AUTOMATIONWORX

User Manual

UM EN ILC 330/350

Order No.: 2699370

Installing and Operating the ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M and ILC 350 PN Inline Controllers

PHOENIX CONTACT
INSPIRING INNOVATIONS

AUTOMATIONWORX

User Manual

Installing and Operating the ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M and ILC 350 PN Inline Controllers

03/2007

Designation: UM EN ILC 330/350

Revision: 05

Order No.: 2699370

This user manual is valid for:

Designation	From Revision (HW)	From Revision (FW)	Order No.
ILC 330 ETH	01	4.6F/1.41	2737193
ILC 350 ETH	01	4.6F/1.11	2737203
ILC 350 ETH/M	02	4.6F/1.20	2985819
ILC 350 PN	02	4.6F/1.3F	2876928

Please Observe the Following Notes

In order to ensure the safe use of the product described, we recommend that you read this manual carefully. The following notes provide information on how to use this manual.

User Group of This Manual

The use of products described in this manual is oriented exclusively to qualified electricians or persons instructed by them, who are familiar with applicable standards and other regulations regarding electrical engineering and, in particular, the relevant safety concepts.

Phoenix Contact accepts no liability for erroneous handling or damage to products from Phoenix Contact or third-party products resulting from disregard of information contained in this manual.

Explanation of Symbols Used

The *attention* symbol refers to an operating procedure which, if not carefully followed, could result in damage to hardware and software or personal injury.

The *note* symbol informs you of conditions that must be strictly observed to achieve error-free operation. It also gives you tips and advice on the efficient use of hardware and on software optimization to save you extra work.

The *text* symbol refers to detailed sources of information (manuals, data sheets, literature, etc.) on the subject matter, product, etc. This text also provides helpful information for the orientation in the manual.

We Are Interested in Your Opinion

We are constantly striving to improve the quality of our manuals.

Should you have any suggestions or recommendations for improvement of the contents and layout of our manuals, please send us your comments.

PHOENIX CONTACT GmbH & Co. KG
Documentation Services
32823 Blomberg
Germany

Phone +49 - 52 35 - 30 0
Fax + 49 - 52 35 - 34 20 21
E-mail tecdoc@phoenixcontact.com

General Terms and Conditions of Use for Technical Documentation

Phoenix Contact GmbH & Co. KG reserves the right to alter, correct, and/or improve the technical documentation and the products described in the technical documentation at its own discretion and without giving prior notice, insofar as this is reasonable for the user. The same applies to any technical changes that serve the purpose of technical progress.

The receipt of technical documentation (in particular data sheets, installation instructions, manuals, etc.) does not constitute any further duty on the part of
Phoenix Contact GmbH & Co. KG to furnish information on alterations to products and/or technical documentation. Any other agreement shall only apply if expressly confirmed in writing by Phoenix Contact GmbH & Co. KG.
Please note that the supplied documentation is product-specific documentation only and that you are responsible for checking the suitability and intended use of the products in your specific application, in particular with regard to observing the applicable standards and regulations.
Although Phoenix Contact GmbH & Co. KG makes every effort to ensure that the information content is accurate, up-to-date, and state-of-the-art, technical inaccuracies and/or printing errors in the information cannot be ruled out. Phoenix Contact GmbH & Co. KG does not offer any guarantees as to the reliability, accuracy or completeness of the information.
All information made available in the technical data is supplied without any accompanying guarantee, whether expressly mentioned, implied or tacitly assumed. This information does not include any guarantees regarding quality, does not describe any fair marketable quality, and does not make any claims as to quality guarantees or guarantees regarding the suitability for a special purpose.

Phoenix Contact GmbH & Co. KG accepts no liability or responsibility for errors or omissions in the content of the technical documentation (in particular data sheets, installation instructions, manuals, etc.).

The aforementioned limitations of liability and exemptions from liability do not apply, in so far as liability must be assumed, e.g., according to product liability law, in cases of premeditation, gross negligence, on account of loss of life, physical injury or damage to health or on account of the violation of important contractual obligations. Claims for damages for the violation of important contractual obligations are, however, limited to contract-typical, predictable damages, provided there is no premeditation or gross negligence, or that liability is assumed on account of loss of life, physical injury or damage to health. This ruling does not imply a change in the burden of proof to the detriment of the user.

Statement of Legal Authority

This manual, including all illustrations contained herein, is copyright protected. Use of this manual by any third party is forbidden. Reproduction, translation, and public disclosure, as well as electronic and photographic archiving or alteration requires the express written consent of Phoenix Contact. Violators are liable for damages.

Phoenix Contact reserves all rights in the case of patent award or listing of a registered design. Third-party products are always named without reference to patent rights. The existence of such rights shall not be excluded.

Internet

Up-to-date information on Phoenix Contact products can be found on the Internet at:

www.phoenixcontact.com

Make sure you always use the latest documentation.
It can be downloaded at:

www.download.phoenixcontact.com

A conversion table is available on the Internet at:

www.download.phoenixcontact.com/general/7000_en_00.pdf

Table of Contents

1 Preface 1-1

1.1 Purpose of this Manual 1-1

1.2 Hardware and Software Requirements 1-1

2 Description of the Inline Controller 2-1

2.1 General Description of the Inline Controller 2-1

2.2 Possible Fields of Application of the ILC 330 ETH, ILC 350 ETH and the ILC 350 ETH/M 2-2

2.3 Possible Fields of Application of the ILC 350 PN 2-3

2.4 Unpacking the Inline Controller 2-4

2.5 Connection and Operating Elements 2-5

2.6 Diagnostic and Status Indicators 2-6

2.7 Mode Selector Switch 2-8

2.8 Reset Button (Concealed) 2-9

2.9 Parameterization Memory 2-10

2.9.1 Using the Parameterization Memory 2-10

2.9.2 Inserting the Parameterization Memory 2-11

2.9.3 Parameterization Memory and Internet Explorer 2-12

2.9.4 Internet Explorer FTP Function 2-13

2.9.5 Activation/Deactivation of the FTP Server 2-14

2.9.6 Activation/Deactivation of the HTTP Server (ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M) 2-14

2.10 Internal Circuit Diagram 2-15

2.11 Mounting and Removing the Inline Controller 2-16

2.12 Communication Paths 2-20

2.12.1 Ethernet 2-21

2.12.2 Serial PRG Interface (Mini-DIN Female Connector) 2-22

2.13 INTERBUS 2-23

2.13.1 Local Bus 2-23

2.13.2 Remote Bus 2-23

2.14 Power Supply 2-24

2.14.1 Sizing of the Power Supply 2-24

2.14.2 Power Supply Connection 2-25

2.14.3 24 V Segment Supply / 24 V Main Supply 2-27

2.14.4 24 V Segment Supply 2-27

2.14.5 24 V Main Voltage 2-27

2.14.6 24 V ILC Supply 2-27

2.14.7 Jumpers 2-27

2.15 Digital Inputs and Outputs 2-28

3 The Inline Controller Under PC WorX 3-1

3.1 Software Version 3-1

3.2 IP Address 3-1

3.3 Setting the Real-Time Clock 3-1

3.4 Download Changes (ILC 330 ETH / ILC 350 ETH / ILC 350 ETH/M) 3-2

3.5 Function Blocks for Handling Files in the Parameterization Memory 3-3

3.5.1 FILE_OPEN 3-4

3.5.2 FILE_CLOSE 3-6

3.5.3 FILE_READ 3-7

3.5.4 FILE_WRITE 3-9

3.5.5 FILE_REMOVE 3-11

3.5.6 FILE_TELL 3-12

3.5.7 FILE_SEEK 3-13

3.5.8 FILE_NOTIFY 3-14

3.6 Function Blocks for TCP/IP Communication 3-17

3.6.1 IP_CONNECT 3-18

3.6.2 IP_USEND 3-20

3.6.3 IP_URCV 3-21

3.6.4 Error Codes 3-22

4 System Variables and Status Information 4-1

4.1 System Variables 4-1

4.1.1 General Information 4-1

4.1.2 Status Register for Local Digital Inputs and Outputs 4-2

4.1.3 Status Register of an I/O Controller (Only ILC 350 PN) 4-3

4.1.4 INTERBUS Register of the ILC 350 PN 4-3

4.1.5 Diagnostic Status Register 4-4

4.1.6 Diagnostic Parameter Register 4-5

4.1.7 IEC 61131 Runtime System 4-6

4.1.8 Control Processor 4-7

4.1.9 Battery, Real-Time Clock, Memory Card 4-7

4.1.10 Power Supplies 4-8

4.1.11 Ethernet Status 4-8

4.1.12 Mode Selector Switch 4-8

4.1.13 System Time 4-9

4.2 Status Information of a PROFINET IO Device (Only ILC 350 PN) 4-10

5 Technical Data and Ordering Data 5-1

5.1 Technical Data 5-1

5.2 Ordering Data 5-7

5.2.1 Modules 5-7

5.2.2 Accessories 5-7

5.2.3 Software 5-7

5.2.4 Documentation 5-7

A Technical Appendix: Service and Maintenance A-1

A 1 Error Causes and Remedies A-1

A 2 Updating the Inline Controller Firmware A-1

A 3 Connecting Unshielded Cables A-2

B Index B-1

1 Preface

1.1 Purpose of this Manual

This manual helps you to start up and operate the ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M or ILC 350 PN Inline Controller.

1.2 Hardware and Software Requirements

- Inline Controller
- 32 MB or 64 MB parameterization memory
- Connecting cable for connecting the Inline Controller to a PC (RS-232)
- PC WorX automation software in the following versions:

ILC 330 ETH	PC WorX version 5.10 or later Service Pack 1 (Part of the AUTOMATIONWORX Software Suite 2007 1.30 Service Pack 1)
ILC 350 ETH	PC WorX version 3.03 or later When using PC WorX 3.03 it is necessary to upgrade the data for the ILC 350 ETH from the "Upgrade ILC 350 ETH" CD. The CD is supplied with the Inline Controller. Upgrading is not required in later versions.
ILC 350 ETH/M	PC WorX version 3.12 or later Service Pack 1 (Part of the AUTOMATIONWORX Software Suite 2005 1.10)
ILC 350 PN	PC WorX version 5.00 or later (Part of the AUTOMATIONWORX Software Suite 2006 1.20)

For the ordering data for hardware and software and additional documentation, please refer to Section "Technical Data and Ordering Data" on page 5-1.

2 Description of the Inline Controller

2.1 General Description of the Inline Controller

The ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M and ILC 350 PN Inline Controllers have the same appearance and numerous identical functions.

Main differences:

ILC 350 ETH/M: This Inline Controller meets the demands of the Germanischer Lloyd (German Lloyd) and the Lloyds Register.

ILC 350 PN: When operating the ILC 350 PN via the Ethernet interface the PROFINET protocol can additionally be used.

In the following, the term Inline Controller is used in general. Differences are particularly mentioned in the individual sections.

The Inline Controller is a compact controller with integrated Ethernet and INTERBUS connections.

IEC 61131 Control Performance

The Inline Controller is seamlessly configured and programmed according to IEC 61131 using the PC WorX automation software. PC WorX can be used locally on the serial interface or via the network (Ethernet). The powerful processor can be programmed in all five IEC 61131 programming languages and ensures quick control task processing.

Integrated Ethernet Connection

The integrated Ethernet connection (using twisted pair cable) provides for the Ethernet connectivity. Throughout the Ethernet network the Inline Controller can be reached via Ethernet and TCP/IP.

Integrated communication functions enable direct and effective data exchange via Ethernet. The Ethernet TCP/IP protocol is used for universal possibilities to communicate with the Inline Controller. Using the TCP/IP send and receive communication blocks according to the IEC 61131-5 standard, information between Inline Controllers, e.g., necessary coupling variables, can be exchanged via Ethernet. This enables distributed, modular automation solutions to be configured.

When using the INTERBUS OPC server, Inline Controller data is available in the Ethernet network in a standardized format and can be used for the different visualization packages.

When operating the **ILC 350 PN** via the Ethernet interface the PROFINET protocol can additionally be used.

Integrated INTERBUS Connection

The I/O level is connected to the Inline Controller using INTERBUS.
The Inline Controller can also be used as the head of an INTERBUS Inline station. It is possible to connect an Inline local bus as well as a remote bus. In this way you can create a complete INTERBUS system using the Inline Controller as the distributed control system.

2.2 Possible Fields of Application of the ILC 330 ETH, ILC 350 ETH and the ILC 350 ETH/M

The **ILC 330 ETH** / **ILC 350 ETH** / **ILC 350 ETH/M** Inline Controller can be used as a distributed control system of an Inline station, which is connected to an Ethernet system. Only one Inline local bus (Figure 2-1) but also a complete INTERBUS system with maximum configuration as remote bus (Figure 2-2) can be connected to the Inline Controller.

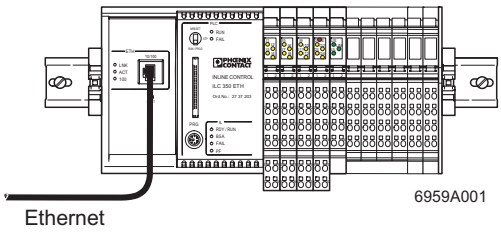


Figure 2-1 Connected Inline local bus

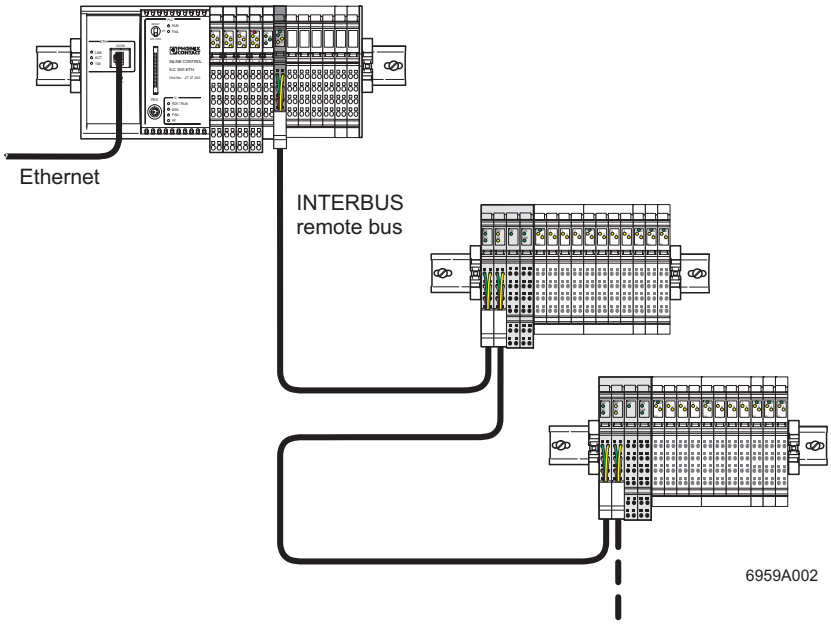


Figure 2-2 Connected remote bus

In an Inline station meeting the GL (German Lloyd) requirements only Inline terminals approved according to the GL must be used. The GL mark is imprinted on the terminals approved according to the German Lloyd. More information on approved terminals can be found on the Internet at www.phoenixcontact.com.

2.3 Possible Fields of Application of the ILC 350 PN

The **ILC 350 PN** Inline Controller can be used as a distributed control system of an Inline station, which is connected to a PROFINET system. Only one Inline local bus (see Figure 2-1) as well as components of a PROFINET network but also a complete INTERBUS system with maximum configuration (see Figure 2-2) can be connected to the ILC 350 PN.

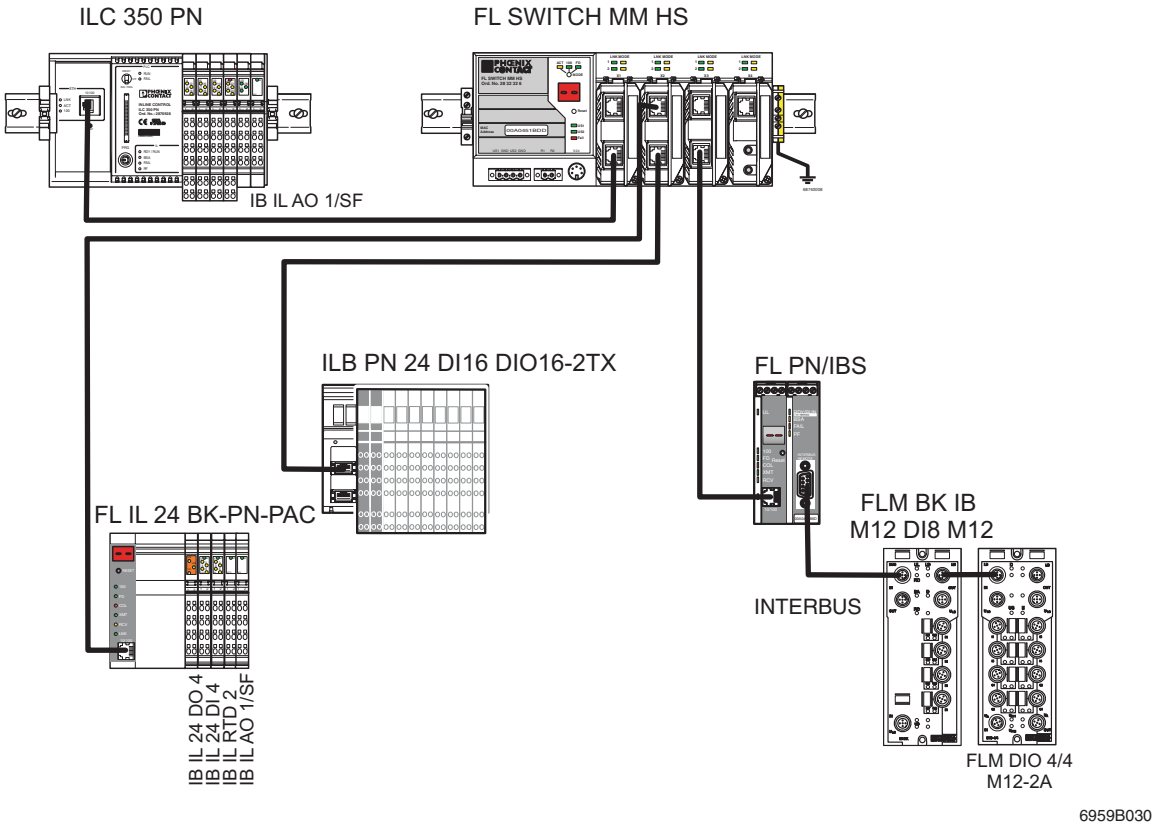


Figure 2-3 ILC 350 PN in a system consisting of PROFINET and INTERBUS

2.4 Unpacking the Inline Controller

The Inline Controller is supplied in an ESD bag together with a package slip with installation instructions. Please read the complete package slip carefully before unpacking the Inline Controller.

Electrostatic discharge

The Inline Controller contains components, which may be damaged or destroyed by electrostatic discharge. Observe the ESD instructions according to EN 61340-5-1 and EN 61340-5-2 when handling the Inline Controller.

To avoid possible damage to the Inline Controller, unpack and pack the controller in accordance with the ESD regulations.

2.5 Connection and Operating Elements

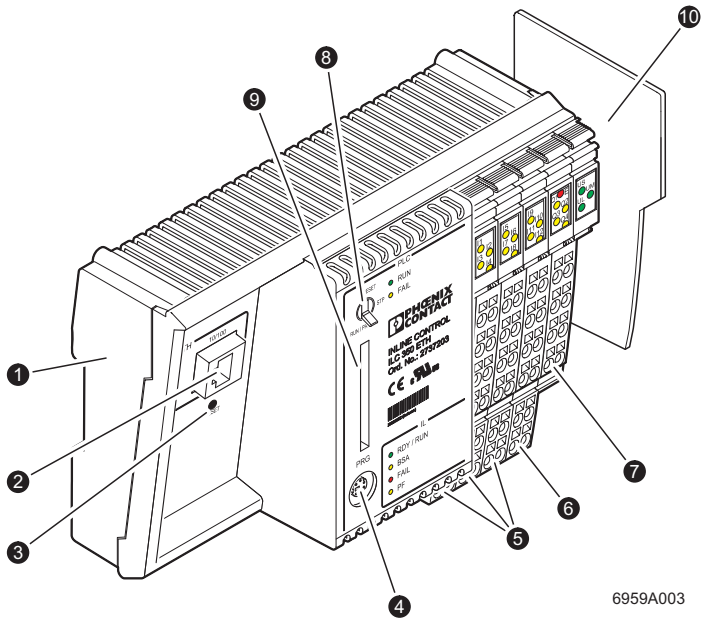


Figure 2-4 Structure of the Inline Controller (example: ILC 350 ETH)

The Inline Controller consists of the following components:

1 Electronics base
2 Ethernet connection
3 Reset button
4 RS-232 interface
5 Connectors 1 to 3: input terminal points
6 Connector 4: output terminal points
7 Connector 5: terminal points for voltage supply
8 Mode selector switch
9 Card holder for parameterization memory
10 End plate

2.6 Diagnostic and Status Indicators

The diagnostic and status indicators are used for quick local error diagnostics.

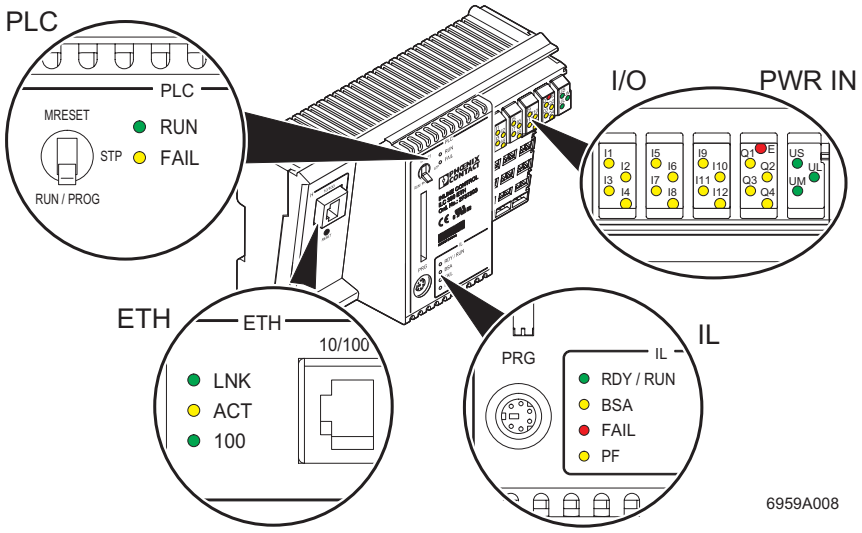


Figure 2-5 Diagnostic and status indicators

Local Diagnostic and Status Indicators

Des.	Color	Meaning
ETH: State of the Ethernet Interface		
LNK	Green	Connection established successfully (link)
	ON:	The Inline Controller is able to contact another network device.
ACT	Yellow	Data transmission (activity)
	ON:	Ethernet interface transmits or receives data
100	Green	Transmission speed
	ON:	100 Mbps
	OFF:	10 Mbps (if LNK is ON)
PLC: Diagnostics of the Inline Controller		
RUN	Green	Inline Controller running
	OFF:	IEC 61131 runtime system not ready to operate
	Flashing:	IEC 61131 runtime system successfully initialized. Control function in READY/STOP state, program not processed.
	ON:	IEC 61131 runtime system successfully initialized and a program is running. Control function in RUN state.
FAIL	Yellow	Failure
	ON:	A runtime error occurred in the IEC 61131 runtime system program.
	OFF:	No runtime error occurred in the IEC 61131 runtime system program.

Des.	Color	Meaning
IL: INTERBUS Diagnostics		
RDY/RUN	Green	INTERBUS master ready to operate / data transmission active (INTERBUS ready/running)
	Flashing:	INTERBUS master in READY or ACTIVE state
	ON:	INTERBUS master in RUN state
BSA	Yellow	Bus segment aborted
	ON:	One or more bus segments in the connected bus are switched off.
FAIL	Red	Failure
	ON:	One of the following errors has occurred: - Bus error in the connected bus (remote bus/local bus) - Controller error
PF	Yellow	Peripheral fault
	ON:	Peripheral fault on a device in the connected bus (local bus or remote bus)
I/O: Digital Inputs and Outputs		
I1 to I12	Yellow	Inputs 1 to 12
	ON:	Corresponding input set
E	Yellow	Error
	ON:	Overload at one of the outputs 1 to 4
Q1 to Q4	Yellow	Outputs 1 to 4
	ON:	Corresponding output set
PWR IN: Supply Voltage (see also Section "Power Supply" on page 2-24)		
US	Green	24 V supply for segment circuit
	ON:	Supply voltage present
UM	Green	24 V supply for main circuit
	ON:	Supply voltage present
UL	Green	24 V supply U_{ILC} for generating the voltages U_L and U_{ANA}
	ON:	Supply voltage present (indication if 24 V supply voltage U_{ILC} present)

2.7 Mode Selector Switch

The mode selector switch is used to define the operating state of the application program.

The RUN/PROG and STP positions have a toggle button function and the MRESET position a pushbutton function. After releasing the button in the MRESET position, it is set back to the STP position.

Table 2-1 Inline Controller operating modes

Operating Mode	Explanation
RUN/PROG	The application program is in the RUN state. The PC WorX software can be used for program and configuration changes. The monitoring and online functions can be used.
STP	The application program is in the STOP state.
MRESET	Retain data and the application program are deleted. For reasons of safety, use the mode selector switch in the following order for deleting the data: • Hold in MRESET position for three seconds. • Release for three seconds. • Hold in MRESET position for three seconds.

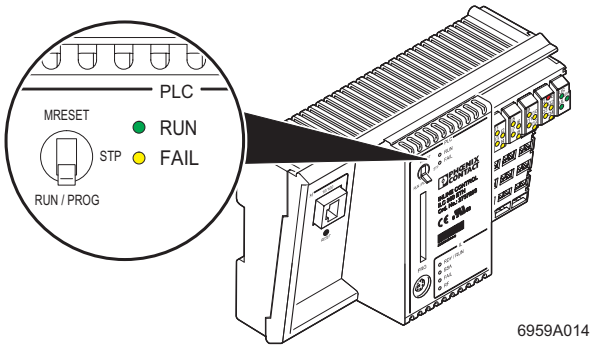


Figure 2-6 Mode selector switch

2.8 Reset Button (Concealed)

The reset button on the Inline Controller can only be operated with a pin. In this way, it is protected against accidental activation. Pressing the reset button causes a complete reset of the Inline Controller.

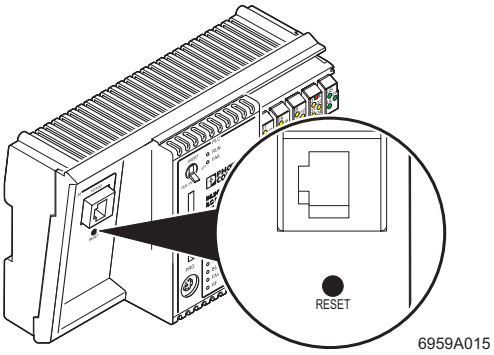


Figure 2-7 Reset button

2.9 Parameterization Memory

2.9.1 Using the Parameterization Memory

A parameterization memory (compact flash card) is essential for operating the Inline Controller. 32-Mbyte and 64-Mbyte parameterization memories are available.

Only use parameterization memories from Phoenix Contact:

– 32-Mbyte parameterization memory
  IBS CF FLASH 32MB Order No. 2737070
– 64-Mbyte parameterization memory
  IBS CF FLASH 64MB Order No. 2737054

Operation without parameterization memory is not permissible.

Only insert and remove the parameterization memory if the supply voltage of the Inline Controller is disconnected.

The parameterization memory must not be removed during Inline Controller operation.

The parameterization memory is recognized during initialization of the Inline Controller. Ensure that the parameterization memory has been inserted before switching on the Inline Controller to enable the Inline Controller to use it.

Retentive (resident) memory blocks are used for the parameterization memory. Therefore, data remains on the parameterization memory even if it is removed from the Inline Controller.

The programs and configurations for your project as well as user-specific files can be stored on the parameterization memory. See Section "Parameterization Memory and Internet Explorer" on page 2-12.

2.9.2 Inserting the Parameterization Memory

Insert the parameterization memory into the card holder provided with the female connector to the front.

Make sure that when the parameterization memory is pressed lightly, it engages with a click in the card holder.

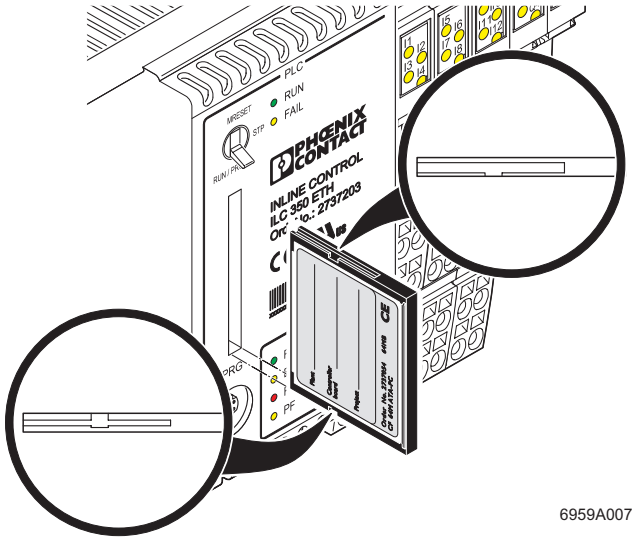


Figure 2-8 Inserting the parameterization memory

One side of the parameterization memory has a labeling field. The file name of your project, for example, can be entered in the labeling field.

2.9.3 Parameterization Memory and Internet Explorer

To delete files or store user-specific files on the parameterization memory, proceed as follows:

The FTP function of Internet Explorer must be activated. See Section "Internet Explorer FTP Function" on page 2-13.

- In PC WorX, switch to the "Bus Configuration" workspace.
- Select the "# ILC 350 ETH 0.0" control system in the "Bus Structure" window.
- Select the "Extended Settings" tab in the "Device Details" window.
- Open Internet Explorer from this window by clicking on the "Open FTP folder on device" button.

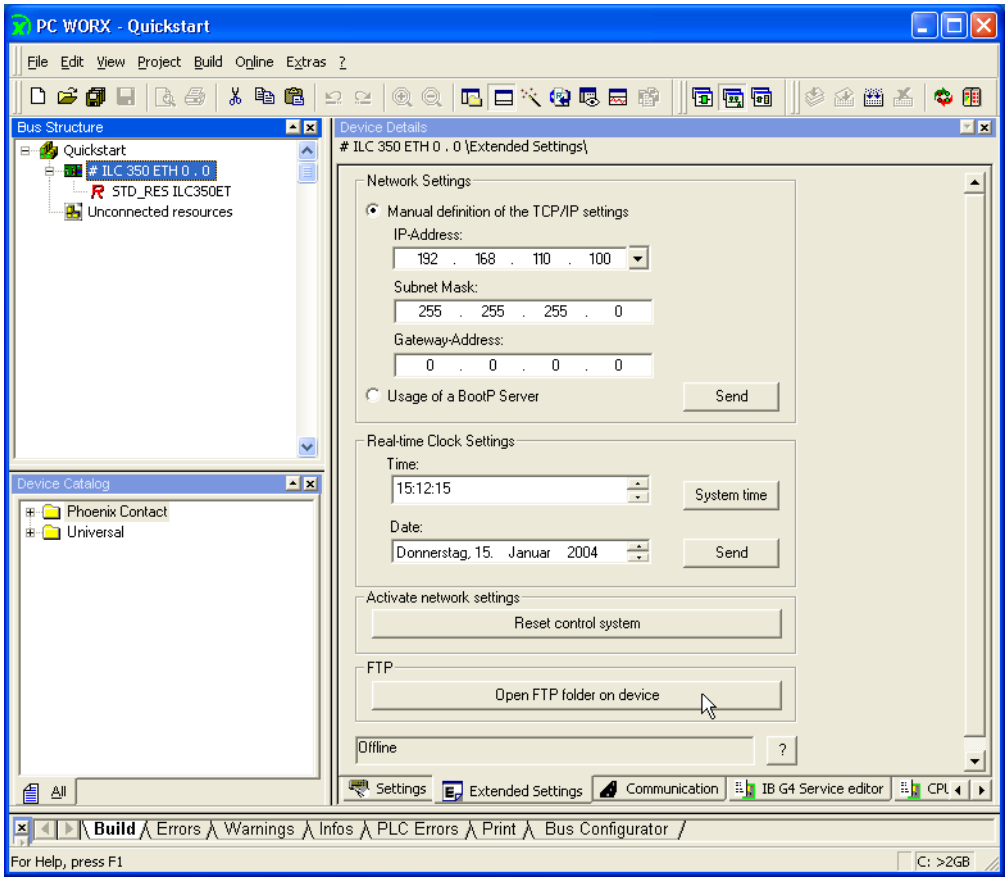


Figure 2-9 Extended Settings: Open FTP folder on device

The file structure, which is stored on the parameterization memory, is displayed in the Internet Explorer window.

Data may only be copied or deleted on the parameterization memory. Do not edit any files as Internet Explorer does not store modified data.

For the current state to be displayed, update the display after every action by means of the "View...Update" command.

2.9.4 Internet Explorer FTP Function

- Enable these settings in Internet Explorer using the "Options...Internet Properties...Advanced" command.

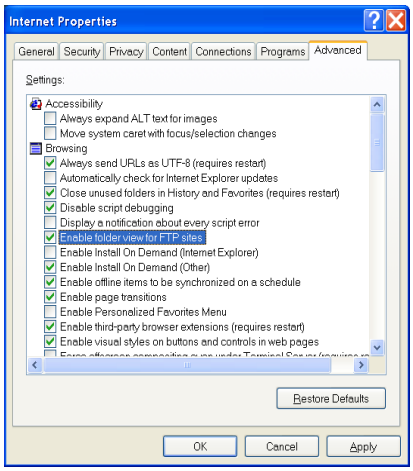


Figure 2-10 Internet Properties: Enable folder view for FTP sites

2.9.5 Activation/Deactivation of the FTP Server

To protect the Inline Controller against unauthorized access, it may be necessary to deactivate the FTP server. The CPU_Set_Value_Request service with Var ID 0172_{hex} is available for this purpose. This service activates or deactivates the FTP server during runtime of the firmware. The FTP server state set is stored retentively and mapped to the ETH_SRV_FTP_ACTIVE system variable. These settings are restored when the Inline Controller is rebooted again.

Value range for the CPU_Set_Value_Request service:

Var ID	0172_{hex}	
Value	0000_{hex}	Deactivate FTP server
	0001_{hex}	Activate FTP server

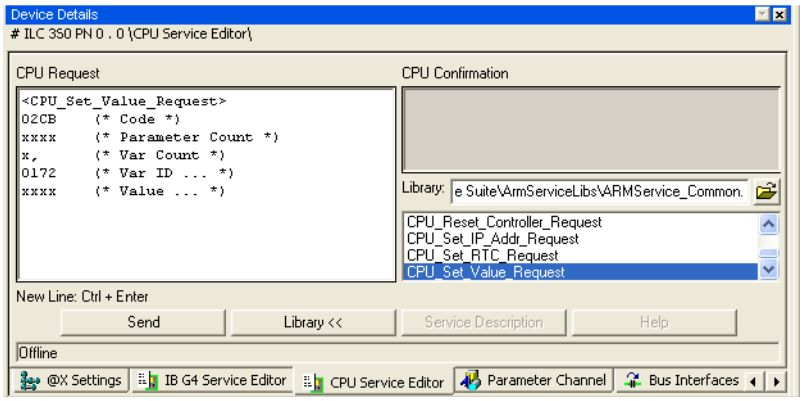


Figure 2-11 Deactivation of the FTP server

2.9.6 Activation/Deactivation of the HTTP Server (ILC 330 ETH, ILC 350 ETH, ILC 350 ETH/M)

To protect the ILC 330 ETH, ILC 350 ETH or ILC 350 ETH/M Inline Controller against unauthorized access, it may be necessary to deactivate the HTTP server. The CPU_Set_Value_Request service with Var ID 0173_{hex} is available for this purpose. This service activates or deactivates the HTTP server during runtime of the firmware. The HTTP server state set is stored retentively and mapped to the ETH_SRV_HTTP_ACTIVE system variable. These settings are restored when the Inline Controller is rebooted again.

Value range for the CPU_Set_Value_Request service:

Var ID	0173_{hex}	
Value	0000_{hex}	Deactivate HTTP server
	0001_{hex}	Activate HTTP server

2.10 Internal Circuit Diagram

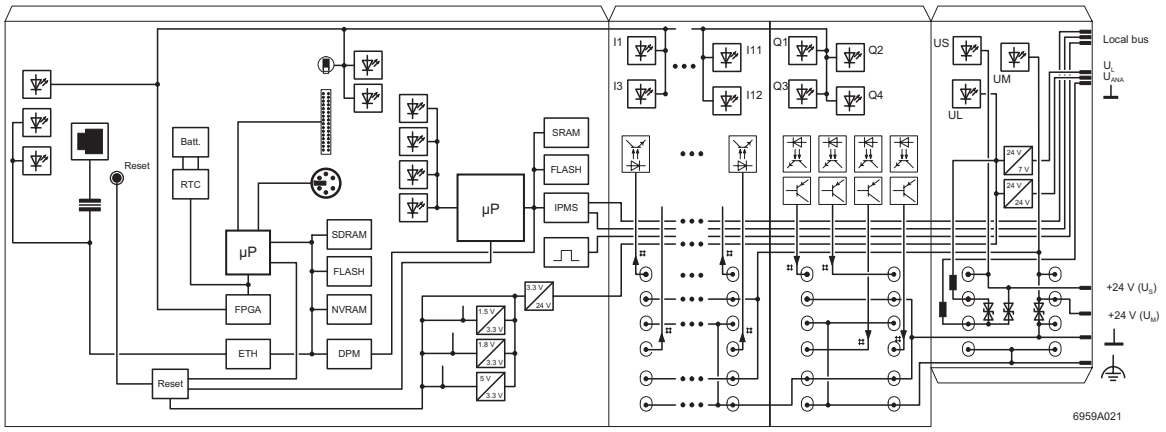


Figure 2-12 Internal circuit diagram

Key:

	LED	FLASH	Non-volatile read/write memory
	RJ45 female connector	NVRAM	Non-volatile RAM
	Transmitter	DPM	Dual-port memory
Reset	Reset button		Mode selector switch
Reset	Reset block		Parameterization memory port
Batt.	Battery		RS-232 interface
RTC	Real-time clock	SRAM	Static RAM
µP	Microprocessor	IPMS	Bus master (IPMS)
FPGA	Field-programmable gate array		Clock generator for local bus clock
ETH	Ethernet		Converter
SDRAM	Synchronous dynamic RAM		
	Optocoupler		Digital output
	Transistor		Resistor
	Digital input		Protective circuit

Other symbols used are explained in the IB IL SYS PRO UM E user manual.

2.11 Mounting and Removing the Inline Controller

For notes and instructions on mounting and removing Inline terminals, please refer to the IB IL SYS PRO UM E user manual (for INTERBUS), the IL SYS INST UM E Inline installation manual or the Inline system manual for your bus system.

Before mounting or removing the Inline Controller, ensure that the power supply is switched off and cannot be switched on again by unauthorized persons.

An Inline station is set up by mounting the individual components side by side. No tools are required. Mounting the components side by side automatically creates potential and bus signal connections between the individual station components.

The modules are mounted perpendicular to the DIN rail.

Mounting location

As all other terminals of the Inline product range the Inline Controller has IP20 protection and is designed for use in a closed control cabinet or control box (terminal box) with IP54 protection or higher.

DIN rail

The Inline Controller is mounted on a 35 mm standard DIN rail.

Fix the DIN rail on which the Inline Controller is mounted several times, especially in the area around the Inline Controller. This makes it easier to remove the Inline Controller.

End clamp

Mount end clamps on both sides of the Inline station. The end clamps ensure that the Inline station is correctly mounted. End clamps fix the Inline station on both sides and keep it from moving from side to side on the DIN rail. Phoenix Contact recommends using CLIPFIX 35-5 end clamps (Order No. 3022276).

End plate

The mechanical end of an Inline station is the end plate. It has no electrical function. It protects the station against ESD pulses and the user against dangerous contact voltage. The end plate is supplied together with the Inline Controller and does not need to be ordered separately.

When mounting or removing, the Inline Controller must be tilted. This means that no Inline terminal must be installed directly to the right of the Inline Controller during mounting and removing. The terminal must be removed prior to mounting or removing the Inline Controller. Otherwise, the locking clips and jumper contacts are damaged.

Mounting position

Mount the Inline Controller horizontally (as shown in Figure 2-13). The specified temperature range (see "Ambient Conditions" on page 5-6) is only guaranteed if the Inline Controller is mounted in this position.

Mounting

When mounting the Inline Controller, proceed as shown in Figure 2-13:

- Disconnect the power to the station.
- Place the Inline Controller onto the rail from above (detail A) and press down firmly (detail B).
- Then attach all the electronics bases required to build the station. Observe the information given in the above user manuals.

Ensure that **all** featherkeys and keyways on adjacent terminals are interlocked.

- Once all the bases have been snapped on, plug the connector into the appropriate base.
  First, place the front connector shaft latching in the front snap-on mechanism (detail C1).
  Then press the top of the connector towards the base until it snaps into the back snap-on mechanism (detail C2).

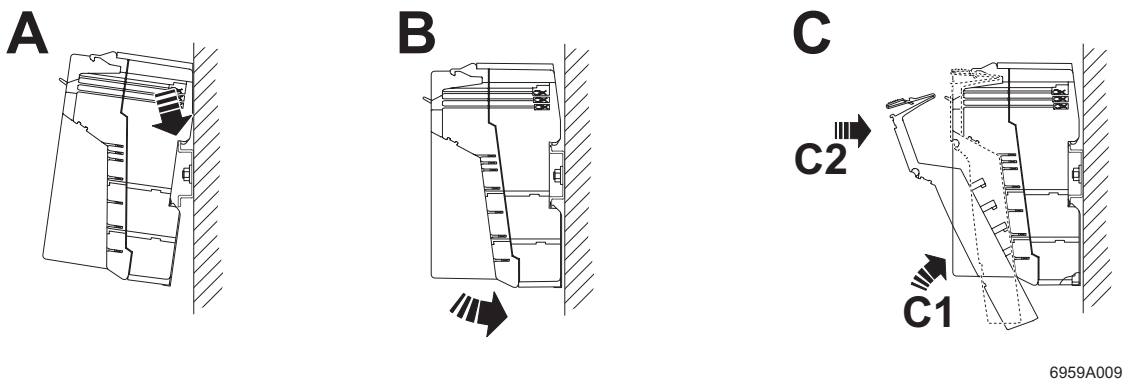


Figure 2-13 Snapping on the Inline Controller

Removing

When removing the Inline Controller, proceed as shown in Figure 2-15:

- Disconnect the power to the station.

Unlike other Inline terminals the Inline Controller is removed by swiveling it from the DIN rail. This requires to remove the Inline terminal to the right prior to removing the Inline Controller. The right connector of the Inline Controller must also be removed.

Remove the third and fourth connector to access the right base latch.

It is therefore recommended to remove all connectors prior to removing the Inline Controller.

- Remove all the connectors of the Inline Controller.
  - Lever up each connector by pressing on the back connector shaft latching (Figure 2-15, detail A1).

When using extended double signal connectors (long connectors), please ensure that you do not bend the connectors back too far, as you may break the front snap-on mechanism.

  - Remove the connectors (Figure 2-15, detail A2).
- If Inline terminals are installed next to the Inline Controller (see Figure 2-14):
  - Remove the following connectors:
    - All connectors of the terminal directly connected (A1 to A4)
    - The adjacent connector of the following terminal (B1)
  - Remove the directly adjacent Inline terminal (A).

Observe the information given in the above user manuals.

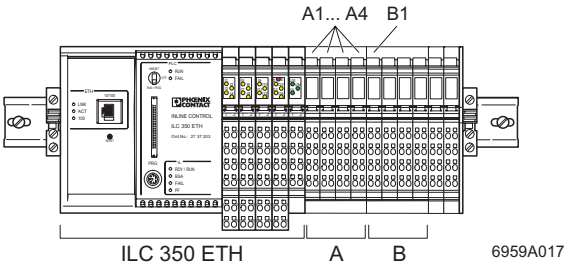


Figure 2-14 Connectors to be removed if terminals are installed next to the Inline Controller

- Insert a tool in the base latches of the Inline Controller one after the other and pull gently upward (Figure 2-15, detail B). Pull out the Inline Controller from the rail (details C1, C2).

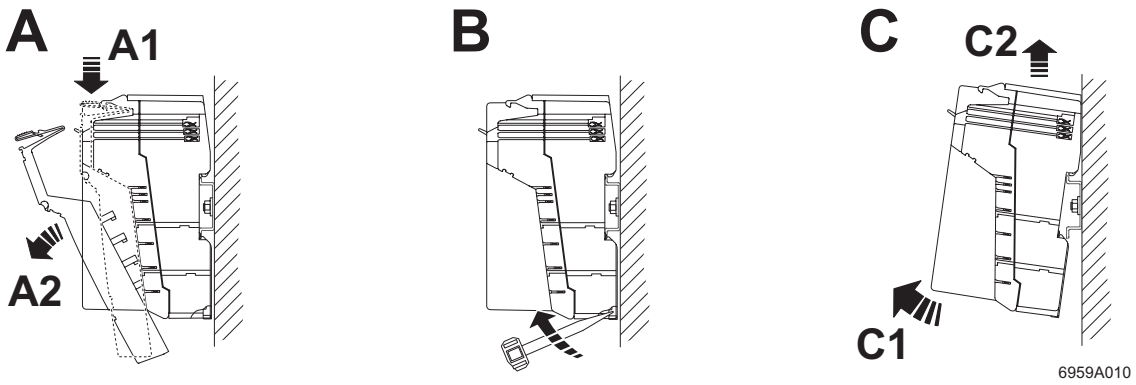


Figure 2-15 Removing the Inline Controller

Replacing the Inline Controller

If you want to replace an Inline Controller within an Inline station, proceed as described above (removing and mounting). Make sure that the terminal to the right is not installed when removing and mounting the Inline Controller. Only reinstall this terminal once the Inline Controller is mounted.

Particularly ensure that **all** featherkeys and keyways on adjacent terminals are interlocked.

Observe the following when replacing an Inline Controller:

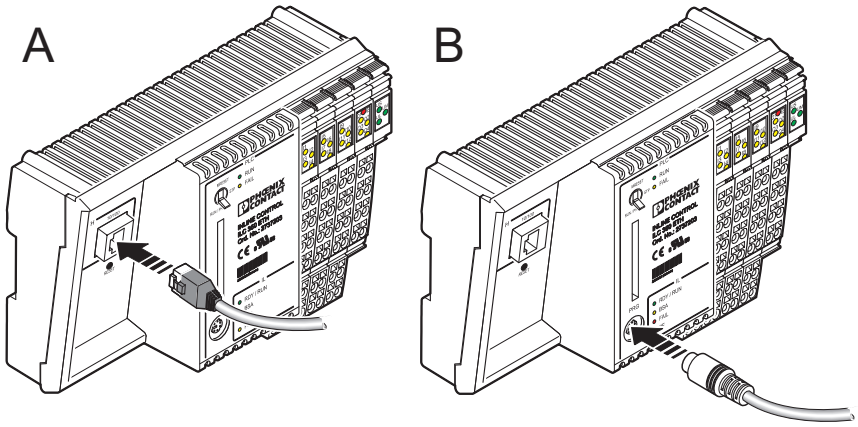
– If you want to use the parameterization settings and projects of the Inline Controller to be replaced, insert its parameterization memory into the new Inline Controller.
– Enter the new MAC address when using the BootP server.

2.12 Communication Paths

The communication path to the Inline Controller must be determined before communication with the Inline Controller can take place.

The following communication paths are available on the Inline Controller:

(A)	Ethernet	10/100Base-T(X)
(B)	PRG	The serial interface (COM 1 to COM 4) of your PC is directly connected to the Inline Controller.



6959A004

Figure 2-16 Communication paths: (A) Ethernet (B) PRG

2.12.1 Ethernet

A standardized Ethernet interface is available for connecting the Ethernet network. The Ethernet network is connected via an RJ45 female connector.

Use an Ethernet cable according to CAT5 of IEEE 802.3.

Planning and installation of your Ethernet network must be carried out according to the IAONA "Industrial Ethernet Planning and Installation Guide" (see Internet at www.iaona-eu.com).

The pin assignment of the interface is as follows:

Transmit data +	T +	1
Transmit data -	T -	2
Receive data +	R +	3
-		4
-		5
Receive data -	R -	6
-		7
-		8

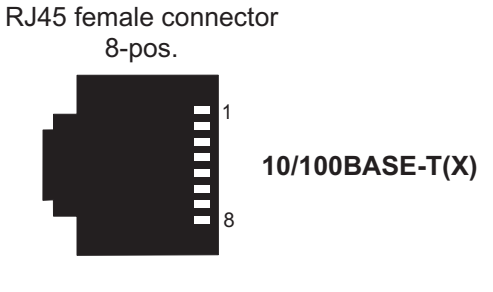


Figure 2-17 Ethernet interface

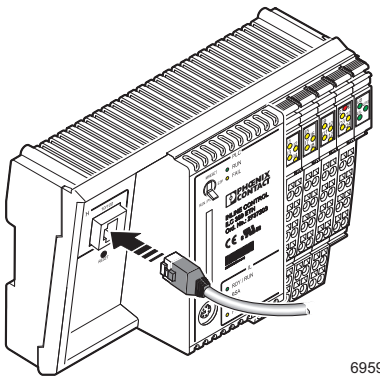


Figure 2-18 Connecting the Ethernet cable to the Inline Controller

A crossover twisted-pair cable is required for direct connection of the Inline Controller to another Inline Controller or to a PC.

2.12.2 Serial PRG Interface (Mini-DIN Female Connector)

This serial interface (also: programming interface) providing the Ethernet interface enables communication with the Inline Controller from a PC. A connecting cable is required for direct connection of the Inline Controller to a PC with PC WorX 3. Connect the connecting cable to the programming interface of the Inline Controller (designation "PRG") and the serial interface of the PC (COM1 to COM4).

Connection

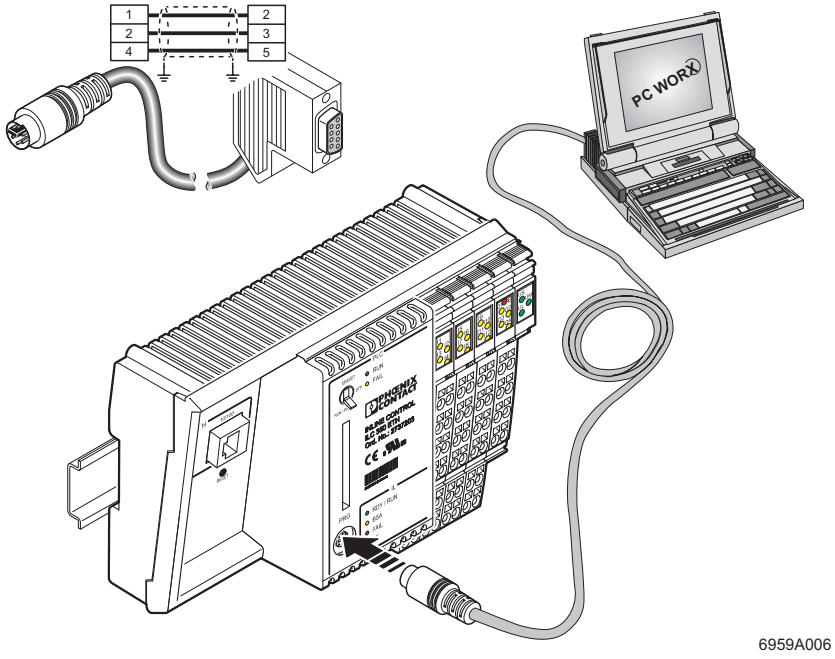


Figure 2-19 Connecting cable between PC and Inline Controller

Ordering data:
Connecting cable for connecting the Inline Controller to a PC (RS-232) for PC WorX, length 3 meters (Order Des. PRG CAB MINI DIN, Order No. 2730611).

2.13 INTERBUS

Observe the information in the "Configuring and Installing the INTERBUS Inline Product Range" user manual IB IL SYS PRO UM E when creating an Inline system (local bus and/or remote bus).

2.13.1 Local Bus

The local bus is automatically created by directly connecting I/O modules to the Inline Controller.

2.13.2 Remote Bus

Connect the remote bus to the Inline Controller using one of the following branch terminals. They only differ in the scope of supply.

– IBS IL 24 RB-T (Order No. 2727941)
– IBS IL 24 RB-T-PAC (Order No. 2861441; including accessories)

The **first branch terminal** must be placed directly behind the Inline Controller. In terms of topology, it opens a **remote bus**.

If **additional branch terminals** are used behind the first branch terminal, they must be installed directly one after the other (see also notes in the terminal-specific data sheet). In terms of topology, the additional branches are **remote bus branches** with the branch terminal being the first device in the corresponding remote bus branch.

Note that isolated connection and disconnection is only possible with remote bus branch interfaces and local bus interfaces. Isolated connection and disconnection is not possible with the remote bus.

The connected remote bus as well as the connected remote bus branches may have further branches. Up to 16 bus levels (i.e., 15 branches) are permitted.

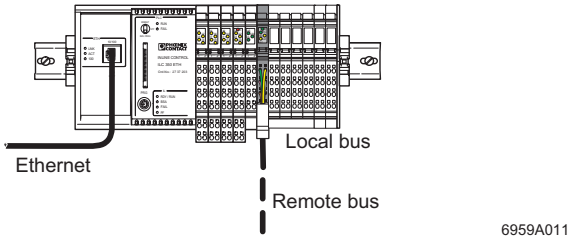


Figure 2-20 Bus configuration with remote bus and local bus

2.14 Power Supply

2.14.1 Sizing of the Power Supply

Use a power supply unit suitable for the currents in your application. The selection depends on the bus configuration, the resulting maximum currents and the type of supply (separate supply of U_{ILC}, U_M and U_S, or supply from one power supply unit).

A **power supply without a fall-back characteristic curve** must be used for correct operation of the Inline Controller (see Figure 2-22).
When the Inline Controller is switched on, an increased inrush current is temporarily triggered. The Inline Controller behaves like a capacitive load when it is switched on.

Some electronically controlled power supplies have a fall-back characteristic curve (see Figure 2-21). They are not suitable for operation with capacitive loads.

A primary switched-mode power supply unit (without fall-back characteristic curve) from the QUINT POWER range (see INTERFACE catalog from Phoenix Contact) is recommended for Inline Controller operation.

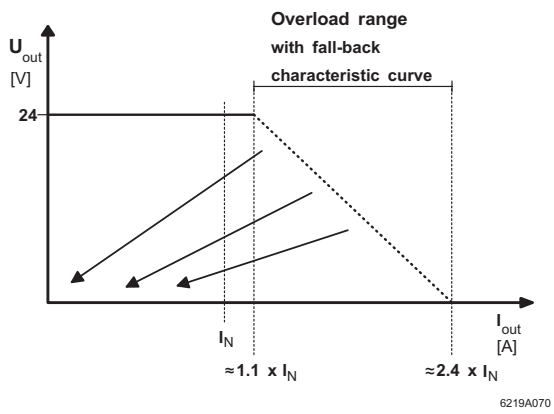


Figure 2-21 Overload range **with** fall-back characteristic curve

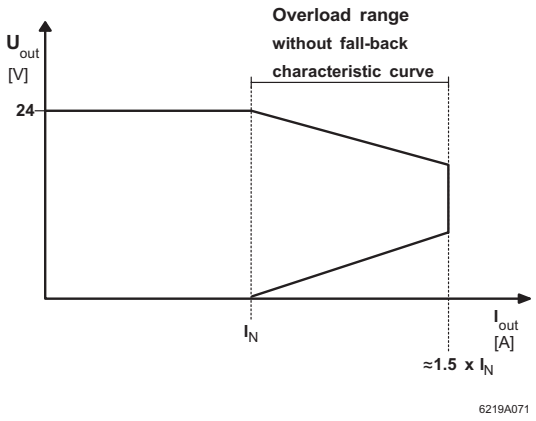


Figure 2-22 Overload range **without** fall-back characteristic curve

2.14.2 Power Supply Connection

Supply the Inline Controller using external 24 V direct voltage sources. The permissible voltage ranges from 20.4 V DC to 30 V DC (ripple included).
The power consumption of the Inline Controller at 24 V typically is 4.8 W (no local bus devices connected).

Only use power supplies that are suitable for operation with capacitive loads (increased inrush current) (see Section "Sizing of the Power Supply" on page 2-24).

1. Connect the power supply to the connector for power supply as shown in Figure 2-23.
2. Plug the connector onto the Inline Controller.
3. Switch on the power supplies.
4. The UL, UM and US LEDs light up and, after about 10 seconds, the RUN and RDY/RUN LEDs start flashing.

The Inline Controller is now fully initialized.

If the LEDs do not light up or start flashing, there is a serious fault on the Inline Controller. Please contact Phoenix Contact.

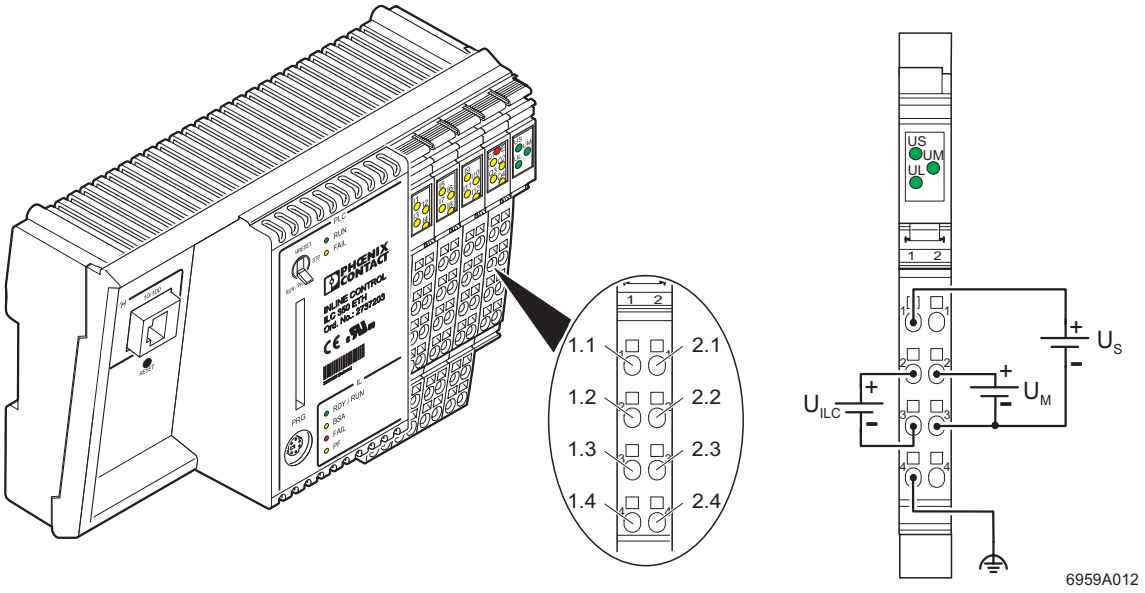


Figure 2-23 Power supply connection

Terminal Point	Assignment		Remark
Connector 5	**Power Connector**		
1.1	24 V DC (U_S)	24 V segment voltage supply	The supplied voltage is directly routed to the potential jumper. Protect the supply voltage externally according to the connected load (local bus devices) with 8 A, maximum. Make sure the external fuse blows in any case.
1.2	24 V DC (U_{ILC})	24 V supply	The 7.5 V communications power (U_L) for the ILC and the connected local bus devices is generated from this voltage. The 24 V analog power (U_{ANA}) for the local bus devices is also generated. Protect the supply voltage externally according to the connected load (local bus devices) with 2 A, maximum. Make sure the external fuse blows in any case.
2.1, 2.2	24 V DC (U_M)	24 V main voltage supply (main power)	The main voltage is routed to the local bus devices via the potential jumpers. Protect the supply voltage externally according to the connected load (local bus devices) with 8 A, maximum. Make sure the external fuse blows in any case.
1.3	LGND	Reference potential for logic ground	The potential is the reference ground for the communications power.
2.3	SGND	Reference potential for segment ground	The reference potential is directly forwarded to the potential jumper and is, at the same time, ground reference for the main and segment supply.
1.4, 2.4	FE	Functional earth ground (FE)	Functional earth ground must be connected through the power supply. The contacts are directly connected to the potential jumper and FE springs on the bottom of the housing. The Inline Controller is grounded when it is snapped onto a grounded DIN rail. Functional earth ground is only used to discharge interference.

The maximum total current flowing through the potential jumpers is 8 A.

2.14.3 24 V Segment Supply / 24 V Main Supply

The segment supply and main supply must have the same reference potential. An electrically isolated architecture is not possible.

2.14.4 24 V Segment Supply

There are several ways of providing the segment voltage at connector 5:

1. You can provide the segment voltage separately at terminal points 1.1 and 2.3 (GND) (see Figure 2-23 on page 2-25).
2. You can jumper connections 1.1 and 2.1 (or 2.2) to ensure that the segment circuit is supplied from the main circuit.
3. You can create a switched segment circuit with a switch between terminal points 1.1 and 2.1 (or 2.2).

The 24 V segment supply has protection against polarity reversal and surge voltage.

It does not have short-circuit protection.

The user must provide short-circuit protection. The rating of the preconnected fuse must be such that the maximum permissible load current of 8 A is not exceeded (total current at U_M and U_S).

2.14.5 24 V Main Voltage

The 24 V main supply has protection against polarity reversal and surge voltage.

It does not have short-circuit protection.

The user must provide short-circuit protection. The rating of the preconnected fuse must be such that the maximum permissible load current of 8 A is not exceeded (total current at U_M and U_S).

2.14.6 24 V ILC Supply

The 24 V ILC supply has protection against polarity reversal and surge voltage. These protective elements are only used to protect the power supply unit.

The rating of the preconnected fuse must be such that the maximum permissible load current of 2 A is not exceeded.

2.14.7 Jumpers

Terminals 1.3 and 2.3 on connector 5 can be jumpered if the communications power and the segment power are not to be electrically isolated.

2.15 Digital Inputs and Outputs

Twelve 24 V DC inputs are available. Eight of the twelve inputs are fast inputs (high-speed I/Os) (from version 01/4.6F/1.11 or later). In addition, twelve 24 V DC outputs are available.

The fast inputs I1 to I8 locally ensure shortest response times. The fast inputs can be used as interrupt inputs. If a defined edge change occurs at one of the inputs, an IEC 61131 freely programmable event task can be called.

Inputs I9 to I12 are standard inputs.

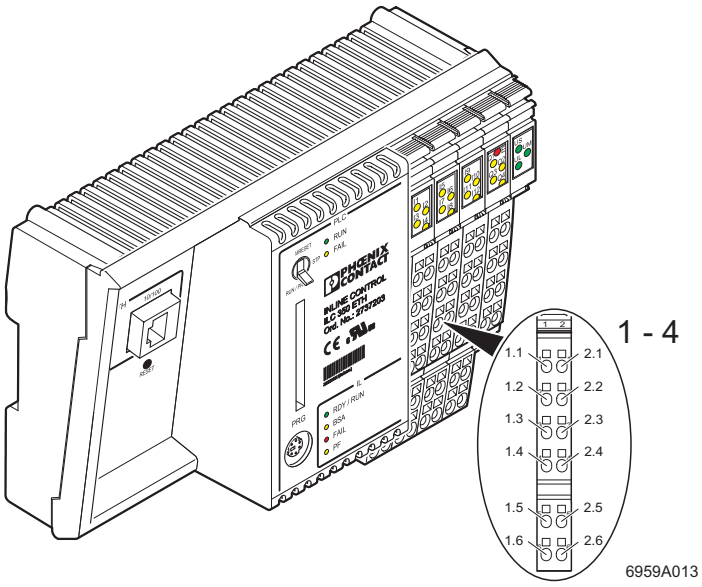


Figure 2-24 Assignment of the terminal points of connectors 1 to 4

Table 2-2 Terminal point assignment

Terminal Point	Assignment	Remark
Connector 1	**Input Terminal Points**	
1.1	I1	Input 1
2.1	I2	Input 2
1.2, 2.2	24 V	Supply voltage U_M for 2 and 3-wire termination
1.3, 2.3	GND	Ground contact for 3-wire termination
1.4	I3	Input 3
2.4	I4	Input 4
1.5, 2.5	24 V	Supply voltage U_M for 2 and 3-wire termination
1.6, 2.6	GND	Ground contact for 3-wire termination
Connector 2	**Input Terminal Points**	
1.1	I5	Input 5
2.1	I6	Input 6
1.2, 2.2	24 V	Supply voltage U_M for 2 and 3-wire termination
1.3, 2.3	GND	Ground contact for 3-wire termination
1.4	I7	Input 7
2.4	I8	Input 8
1.5, 2.5	24 V	Supply voltage U_M for 2 and 3-wire termination
1.6, 2.6	GND	Ground contact for 3-wire termination
Connector 3	**Input Terminal Points**	
1.1	I9	Input 9
2.1	I10	Input 10
1.2, 2.2	24 V	Supply voltage U_M for 2 and 3-wire termination
1.3, 2.3	GND	Ground contact for 3-wire termination
1.4	I11	Input 11
2.4	I12	Input 12
1.5, 2.5	24 V	Supply voltage U_M for 2 and 3-wire termination
1.6, 2.6	GND	Ground contact for 3-wire termination

The inputs are supplied with 24 V DC from the main supply (U_M).

Table 2-2 Terminal point assignment (Forts.)

Terminal Point	Assignment	Remark
Connector 4	**Output Terminal Points**	
1.1	O1	Output 1
2.1	O2	Output 2
1.2, 2.2	GND	Ground contact for 2 and 3-wire termination
1.3, 2.3	FE	Functional earth ground for 3-wire termination
1.4	O3	Output 3
2.4	O4	Output 4
1.5, 2.5	GND	Ground contact for 2 and 3-wire termination
1.6, 2.6	FE	Functional earth ground for 3-wire termination

The outputs are supplied with 24 V DC from the segment supply (U_S).

The outputs have protection when the ground connection is interrupted and must be wired accordingly.

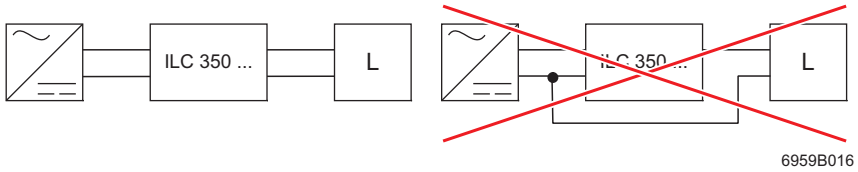


Figure 2-25 Basic wiring of an output with a load (L)

3 The Inline Controller Under PC WorX

3.1 Software Version

Using the Inline Controllers requires the following PC WorX version:

ILC 330 ETH	PC WorX version 5.10 or later Service Pack 1 (Part of the AUTOMATIONWORX Software Suite 2007 1.30 Service Pack 1)
ILC 350 ETH	PC WorX version 3.03 or later When using PC WorX 3.03 it is necessary to upgrade the data for the ILC 350 ETH from the "Upgrade ILC 350 ETH" CD. The CD is supplied with the Inline Controller. Upgrading is not required in later versions.
ILC 350 ETH/M	PC WorX version 3.12 or later Service Pack 1 (Part of the AUTOMATIONWORX Software Suite 2005 1.10)
ILC 350 PN	PC WorX version 5.00 or later (Part of the AUTOMATIONWORX Software Suite 2006 1.20)

Information on installing and using PC WorX can be found in the quick start guide for the PC WorX version used. The quick start guide can be downloaded free of charge at www.phoenixcontact.com and is supplied with the software.

3.2 IP Address

BootP is activated by default upon delivery for communication with the Inline Controller via Ethernet.

The first setting of the IP address can additionally be carried out with the PC WorX software manually via the serial interface or by means of a BootP server.

It is possible to change the IP address later via the serial interface or Ethernet using the PC WorX software.

For setting the IP address using PC WorX, please proceed as explained in the quick start guide for the PC WorX version used.

3.3 Setting the Real-Time Clock

Set the time and date for the internal system clock of the Inline Controller using the Controller "Device Details" window in PC WorX.

For setting the real-time clock, please proceed as explained in the quick start guide for the PC WorX version used.

3.4 Download Changes (ILC 330 ETH / ILC 350 ETH / ILC 350 ETH/M)

The ILC 350 ETH Inline Controller version 4.6F/1.20 or later with processor ARM_L_40 and the Inline Controllers ILC 330 ETH and ILC 350 ETH/M support the "Download Changes" function.

The "Download Changes" function can be used to download project modifications to the control system(s) where they can be activated without stopping the running PLC.

For detailed information on this function, please refer to the PC WorX online help.

When downloading changes to one or more control systems, please observe the following:

Specify if the system is to ensure the real-time behavior of the PLC prior to starting the download process.

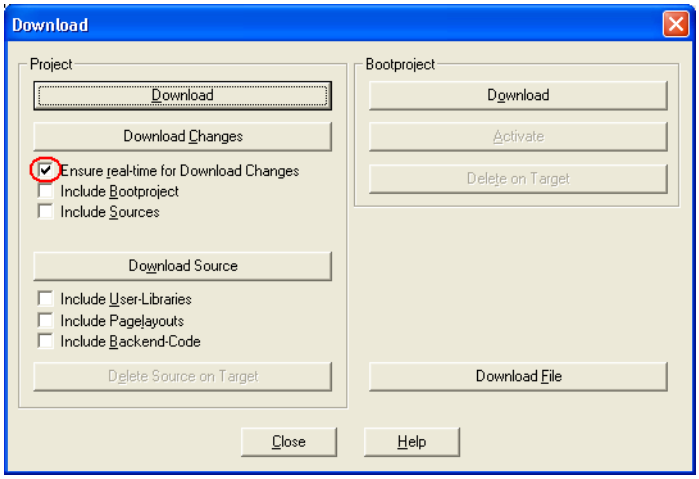


Figure 3-1 Ensure real-time behavior

If the "Ensure real-time for Download Changes" checkbox is enabled in the "Download" dialog box (recommended standard settings), the system will attempt to "Download Changes" without affecting the real-time conditions of the tasks executed on the control system.

If downloading changes is not possible, e.g., the data volume to be copied is too large, an error message is output. In this case, disable the "Ensure real-time..." option (i.e., real-time violations permitted) in order for all changes to be executed. (In this context, please note the background information in the real-time behavior section in the online help.)

If real-time violations are permitted, first ensure that your PLC application is running within safe limits. Note that any real-time violation may lead to unexpected consequences in the automation system as user tasks may not be processed for a short period. Therefore, check the level of risk before disabling this option.

3.5 Function Blocks for Handling Files in the Parameterization Memory

The function blocks are used to access files from within the application program. Some of the blocks support multiple instantiation. This means that it is possible to work with a number of different files within the same project. The blocks perform the standard functions that are required for typical file-access operations.

The FILE_NOTIFY block is available in addition to the blocks for typical file-access operations. This block can be used to detect files that have been modified in a directory containing user files. These modifications may include:

– Deletion of one or more files
– Writing of one or more new files
– Modification of one or more existing files

Both modifications made to this directory via FTP (remote) and modifications made locally via function blocks or firmware services can be detected.

All file operations are subject to the following restrictions:
No directory hierarchies are supported. All file operations only affect the root directory of the parameterization memory.

Table 3-1 Overview of function blocks

Function Block	Short Description
FILE_OPEN	Opens a file with a specific name
FILE_CLOSE	Closes a file with a specific handle
FILE_READ	Reads from a file with a specific handle
FILE_WRITE	Writes to a file with a specific handle
FILE_REMOVE	Deletes a file with a specific name
FILE_TELL	Determines the current position of the file pointer in a file
FILE_SEEK	Moves the current file pointer to a new position
FILE_NOTIFY	Displays files that have recently been created, deleted or modified

Technical Properties (Limit Values)

– The string for a file name (for opening, deleting, and monitoring modifications) must not exceed 80 characters in length.
– Up to 64 files can be monitored for modifications.
– Maximum number of instances for function blocks:
  FILE_NOTIFY: 1 instance
  Others: 8 instances
– Minimum scan time for FILE_NOTIFY: 1 second

3.5.1 FILE_OPEN

The FILE_OPEN block is used to open a file in the parameterization memory with a specific name.

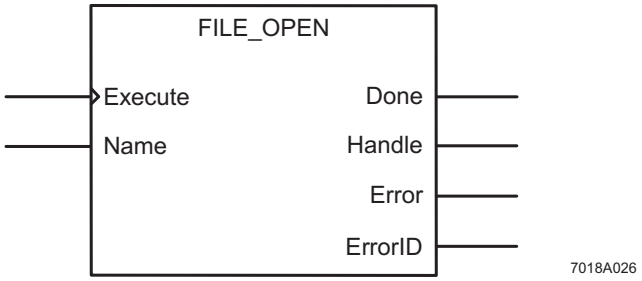


Figure 3-2 FILE_OPEN block

Assign a STRING variable containing the name of the file to be opened to the "Name" input.

Activate the block on a rising edge at the "Execute" input. The file is opened.

If the file to be opened does not exist, it is created on the data carrier. The file is opened with read and write access rights. If "Done" is set to 1 and "Error" = 0, a valid file handle is present at the "Handle" output.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

When using the block, please note:

– This block can activate up to eight instances at the same time. This means that only eight different files can be opened at the same time.
– The same file may not be opened more than once.
– The file to be opened must not be write-protected, because files are always opened with read and write access rights.
– The variable in which the file name is entered must be a STRING data type.
– The file name must not exceed 80 characters in length.
– The status of the "Done", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-2 FILE_OPEN function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block opens or creates a file.
Name	STRING	Input	Name of the file to be opened or created.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Handle	UINT	Output	File handle of the file opened or created.
Error	BOOL	Output	FALSE: No errors occurred when opening/creating the file. TRUE: An error occurred when opening/creating the file.
ErrorID	UINT	Output	Error number of error that occurred when opening/creating the file: 0: No error information available. 2: The maximum number of files is already open. 4: The file is already open. 5: The file is write-protected or access was denied. 6: File name not entered.

3.5.2 FILE_CLOSE

The FILE_CLOSE block is used to close a file in the parameterization memory with a specific handle.

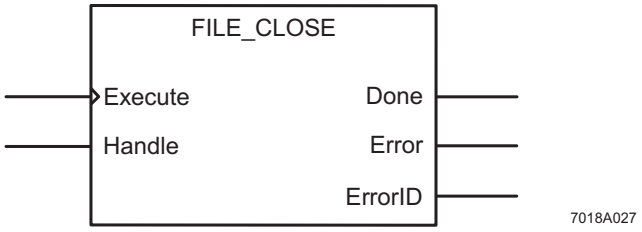


Figure 3-3 FILE_CLOSE block

Assign the valid handle of an open file to the "Handle" input.

Activate the block on a rising edge at the "Execute" input. The file is closed.

Execution of the function block is complete when the "Done" output is set to 1.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-3 FILE_CLOSE function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block closes the file.
Handle	UINT	Input	File handle of file to be closed.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Error	BOOL	Output	FALSE: No errors occurred when closing the file. TRUE: An error occurred when closing the file.
ErrorID	UINT	Output	Error number of error that occurred when closing the file: 0: No error information available. 1: Invalid file handle. 20: File could not be closed.

3.5.3 FILE_READ

The FILE_READ block is used to read data from a file that was opened previously.

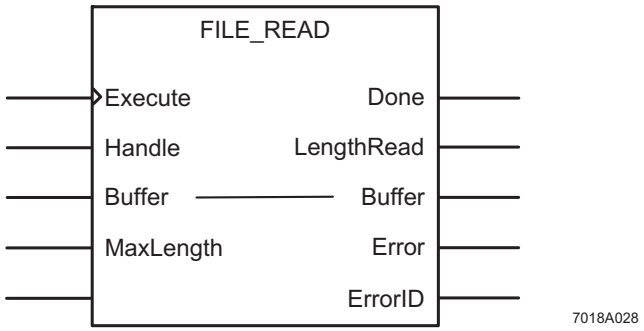


Figure 3-4 FILE_READ block

Assign the valid handle for the file from which data is to be read to the "Handle" input.

Create a data buffer of sufficient size for the "Buffer" output. The data buffer for the data to be read can be declared in a number of ways. The type of data buffer can be defined by the user, e.g., a byte array. Strings cannot be used directly as data buffers. If the data read is to be processed further as a string, it must first be saved in an array and then converted into a string using the BUF_TO_STRING function block.

Enter the number of bytes to be read at the "MaxLength" input.

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

Following successful execution, the data is copied to the buffer and the number of characters read is indicated at the "LengthRead" output. The number of characters read may be smaller than "MaxLength" if the end of the file was reached before the number of characters indicated was processed.

Table 3-4 FILE_READ function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block reads data from file.
Handle	UINT	Input	File handle of file from which data is to be read.
Buffer	ANY	Output	Data buffer for data to be read.
MaxLength	UDINT	Input	Number of characters to be read.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Length Read	UDINT	Output	Number of characters read.
Error	BOOL	Output	FALSE: No errors occurred during reading. TRUE: An error occurred during reading.
ErrorID	UINT	Output	Error number of error that occurred when reading the file: 0: No error information available. 1: Invalid file handle. 10: End of file reached. 12: Number of characters to be read is greater than the data buffer. 22: No data could be read.

3.5.4 FILE_WRITE

The FILE_WRITE block is used to write data to a file that was opened previously.

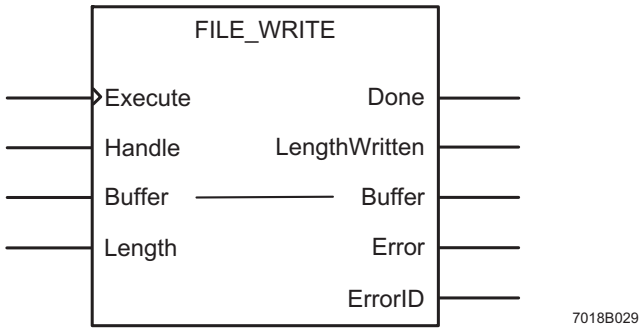


Figure 3-5 FILE_WRITE block

Assign the valid handle for the file to which data is to be written to the "Handle" input.

Create a data buffer of sufficient size, which contains the data to be written, for the "Buffer" input. The data buffer for the data can be declared in a number of ways. The type of data buffer can be defined by the user, e.g., a byte array. Strings cannot be used directly as data buffers. If a string is to be saved in a file, the string must first be saved in an array using the STRING_TO_BUF function block.

Assign the number of bytes to be written to the "Length" input.

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1.

If the function could not be executed successfully, the number of bytes written will be indicated at the "LengthWritten" output. It is the same as the number of characters to be written.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "LengthWritten", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-5 FILE_WRITE function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block writes data to file.
Handle	UINT	Input	File handle of file to which data is to be written.
Buffer	ANY	Output	Data buffer containing data to be written.
Length	UDINT	Input	Number of characters to be written.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Length Written	UDINT	Output	Number of characters written.
Error	BOOL	Output	FALSE: No errors occurred during writing. TRUE: An error occurred during writing.
ErrorID	UINT	Output	Error number of error that occurred when writing: 0: No error information available. 1: Invalid file handle. 11: No memory is available for writing the data. 12: Number of characters to be written is greater than data buffer. 23: No data could be written.

3.5.5 FILE_REMOVE

The FILE_REMOVE block is used to delete a file with a specific name.

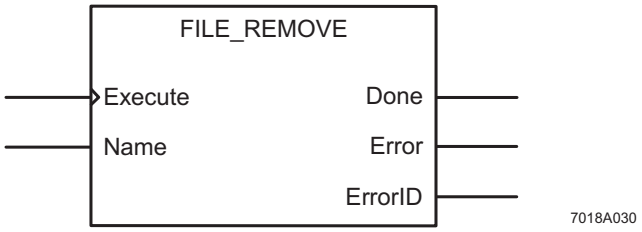


Figure 3-6 FILE_REMOVE block

Before deletion, the file to be deleted must be closed using the FILE_CLOSE function block.

Assign a STRING variable containing the name of the file to be deleted to the "Name" input. The file name must not exceed 80 characters in length.

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-6 FILE_REMOVE function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block deletes the file.
Name	BOOL	Input	Name of the file to be deleted.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Error	BOOL	Output	FALSE: No errors occurred during deleting. TRUE: An error occurred during deleting.
ErrorID	UINT	Output	Error number of error that occurred when deleting the file: 0: No error information available. 2: The maximum number of files is already open. 3: File could not be found. 5: The file is open or write-protected or access was denied. 6: File name not entered. 21: File could not be deleted.

3.5.6 FILE_TELL

The FILE_TELL block is used to determine the current position of the file pointer in a file.

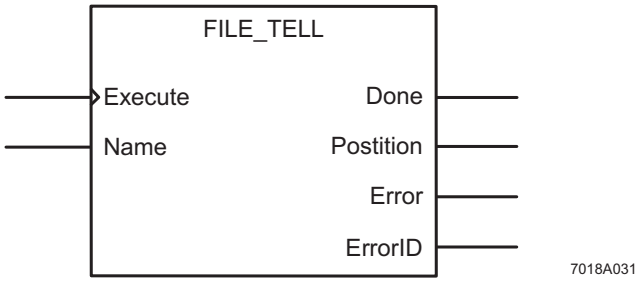


Figure 3-7 FILE_TELL block

In order to determine the current position of the file pointer (relative offset from start of file), the valid handle for the file is assigned to "Handle".

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1.

The current position of the file pointer can be read at the "Position" output following successful execution.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "Position", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-7 FILE_TELL function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block determines the current position of the file pointer.
Handle	UINT	Input	File handle of file whose position is to be determined.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Position	DINT	Output	Current position in the file.
Error	BOOL	Output	FALSE: No errors occurred when determining the position. TRUE: An error occurred when determining the position.
ErrorID	UINT	Output	Error number of the error that occurred when determining the position: 0: No error information available. 1: Invalid file handle.

3.5.7 FILE_SEEK

The FILE_SEEK block is used to move the current file pointer to a new position.

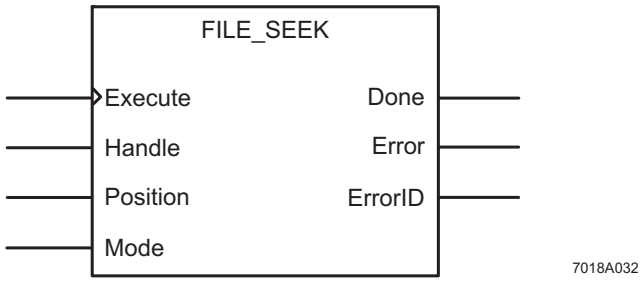


Figure 3-8 FILE_SEEK block

Assign the valid handle of the file to be processed to the "Handle" input. Write the relative offset to be used to move the file pointer to the "Position" input. Enter the reference to be used to move the pointer at the "Mode" input (see table).

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input.

Table 3-8 FILE_SEEK function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: Block sets the new position of the file pointer.
Handle	UINT	Input	File handle of file in which the position is to be set.
Position	DINT	Input	New position in file.
Mode	UINT	Input	Positioning mode 0: Positioning relative to start of file. 1: Positioning relative to current position. 2: Positioning relative to end of file.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Error	BOOL	Output	FALSE: No errors occurred when setting the position. TRUE: An error occurred when setting the position.
ErrorID	UINT	Output	Error number of the error that occurred when setting the position: 0: No error information available. 1: Invalid file handle. 13: Invalid positioning mode. 24: The position could not be set.

3.5.8 FILE_NOTIFY

The FILE_NOTIFY block is used to display the files, which have recently been added, deleted or modified.

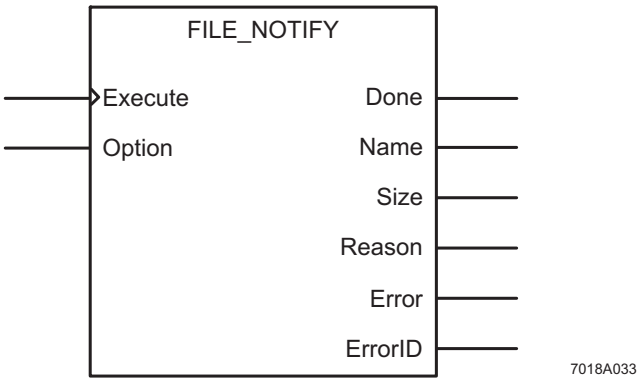


Figure 3-9 FILE_NOTIFY block

Parameterize the block using an option string at the "Option" input. The possible options are indicated in Table 3-10. Only one instance of this block can be used.

Activate the block on a rising edge at the "Execute" input.

Execution of the function block is complete when the "Done" output is set to 1. This is the case if the monitoring condition set at the "Option" input is true.

The name of the file to be modified can be read as a string at the "Name" output.
The size of the file to be modified (in bytes) can be read at the "Size" output.

If an error occurred during execution, the value at the "Error" output = 1.
"ErrorID" indicates the number of the cause of the error.

The status of the "Done", "Error", and "ErrorID" outputs is maintained until a falling edge is detected at the "Execute" input. If the monitoring condition set is true for a number of files, the block is reset as soon as "Done" changes to 1 following a rising edge at "Execute". The next file name can then be read from "Name".

Table 3-9 FILE_NOTIFY function block inputs and outputs

Name	Data Type	Data Direction	Description
Execute	BOOL	Input	Rising edge: The function of the block is activated.
Option	STRING	Input	The operating options are set using this string. The possible options are indicated in Table 3-10. The string must not exceed 80 characters in length.
Done	BOOL	Output	FALSE: The function block was not executed. TRUE: The function block was executed.
Name	STRING	Output	The string pending following the edge at "Done" indicates the file name of a file that has changed. The file name must not exceed 80 characters in length.
Size	UDINT	Output	Size of modified file in bytes. This value can be used, e.g., when reading the file subsequently.
Reason	DWORD	Output	This flag indicates the reason for the modification. Possible reasons include: 1: Addition of a new file. 2: Deletion of an existing file. 4: Modification of an existing file.
Error	BOOL	Output	FALSE: No errors occurred when executing the function. TRUE: An error occurred when executing the function.
ErrorID	UINT	Output	Error number of the error that occurred when executing the function: 0: No error information available. 25: Too many instances open (> 1). 30: Incorrect length of option string (0 or > 80 bytes). 31: Incorrect length of FILENAME (0 or > 80). 32: Incorrect or missing entry for MAXFILES (< 1 or > 64). 33: Incorrect entry for SCANTIME (< 1000 or > 1,000,000). 34: Too many files in directory (> MAXFILES). 35: Block resources conflict (e.g., insufficient memory)

Table 3-10 Options for the option string

Option	Meaning
/MAXFILES	Number of files to be monitored. The permissible value range is between 1 and 64. Memory usage increases along with the number of MAXFILES.
/SCANTIME	Cycle time during which a specific directory is scanned for modifications. Value entered in seconds. The minimum cycle time is 1 second.
/FILENAME	Name of a specific file group to be monitored. This option can be called a number of times to monitor different file names and file extensions. The file name must not exceed 80 characters in length. Only printable characters may be used. The following special characters are relevant for the file name mask: ? Wildcard for a variable character in a constant character string Example: "Test_?_200?.log". * Wildcard for a variable character string in a constant character string Example: "LogFile*". If the /FILENAME option is not used, all file names and extensions will be monitored.

Example for option string

A typical option string may have the following structure:

/MAXFILES=20 /SCANTIME=5000 /FILENAME=test_?.txt /FILENAME=*.log

In this example, up to 20 files can be monitored.
The default directory is scanned every 5 seconds for file modifications.
Files with "log" or "txt" extensions are monitored.
Only modifications to "txt" files starting with the character string "test_" are detected. In addition, all files with the extension "log" are monitored.

Example of possible error

The size of the modified file written to SIZE may not be correct. This may be the case if, for example, a large file has been copied to the parameterization memory via FTP and the FILE_NOTIFY block has been parameterized with a SCAN_TIME of 1 second. In this case, the block immediately indicates that a new file has been created and provides information about the current size of this file. However, as the file is still being written, the size indicated does not tally with the actual size of the file.

3.6 Function Blocks for TCP/IP Communication

The functions blocks are used to implement Ethernet communication between two communication partners. For this reason the previous IEC 61131-5 function blocks have been extended by a TCP/IP mode.

Possible applications include:

– Communication between two Phoenix Contact control systems (e.g., ILC - RFC, ILC - FC, ILC - ILC) and
– Communication between a Phoenix Contact control system and any other TCP/IP-capable device (e.g., ILC - PC, ILC - PLC).

Implement all time and connection monitoring functions in the application program.

The Inline Controller supports 16 TCP/IP connections to another communication partner.

The function blocks are valid for:

Order Designation	From Hardware Version	From Firmware Version
ILC 330 ETH	01	4.6F/1.41
ILC 350 ETH	01	4.6F/1.11
ILC 350 ETH/M	02	4.6F/1.11
ILC 350 PN	02	4.6F/1.3F

Table 3-11 Overview of function blocks

Function Block	Short Description
IP_CONNECT	Connection establishment between two communication partners
IP_USEND	Transmitting data to a communication partner
IP_URCV	Receiving data from a communication partner

A description of the communication blocks can be found in the PC WorX online help. This user manual only describes the extensions of the TCP/IP function blocks.

3.6.1 IP_CONNECT

This block is used to establish a connection between two communication partners via an IP connection.

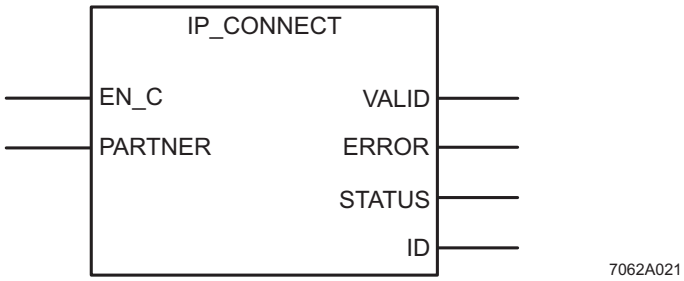


Figure 3-10 IP_CONNECT function block

Table 3-12 IP_CONNECT function block inputs and outputs

Name	Data Type	Data Direction	Description
EN_C	BOOL	Input	When setting this input to TRUE, the block attempts to establish a connection to the IP address specified at the PARTNER input. A connection can only be established if the block is called on both communication partners followed by a communication establishment request of both communication partners. FALSE The connection has been aborted. TRUE The connection has been established or maintained.
PARTNER	STRING	Input	This input specifies the communication partner to be used for communication. For this purpose, enter the partner IP address followed by several optional parameters. Use the syntax given below the table.
VALID	BOOL	Output	This output indicates if a valid communication connection has been established. After connection establishment the output is set to TRUE. The output remains in the TRUE state as long as the connection is established. The return ID of the communication connection must be transferred to the IP_USEND and IP_URCV function blocks. For this purpose, you can directly connect the ID output of the IP_CONNECT function block to the ID input of the IP_USEND function block and to the ID input of the IP_URCV function block.
ERROR	BOOL	Output	Rising edge: An error has occurred. The error code can be read at the STATUS output during the cycle with the rising edge.
STATUS	INT	Output	Error codes of the error occurred.
ID	INT	Output	Unique designation for the communication connection. In the event of several existing communication connections, you can use the ID parameter to distinguish the connections from one another. ID is only valid if output VALID = TRUE.

Syntax for Entering TCP-Based Communication at the PARTNER Input (Example)

Note that the entry is case-sensitive. Enter only one space between the parameters.

If you do not enter the parameters, the default values will be used.

Minimum configuration:

```
/ACTIVE /PORT=502 /IP=172.16.100.35
/PASSIVE /PORT=502
```

Maximum configuration:

```
/ACTIVE /PORT=502 /IP=172.16.100.35 /CONTO=3000 /RCVTO=5000
/PASSIVE /PORT=502 /IP=172.16.100.35 /CONTO=3000 /RCVTO=5000
```

When using TCP/IP connections between a control system and another TCP/IP-capable device, one partner must be active and the other partner must be passive.

Parameters:

Parameter	Meaning
PASSIVE	If EN_C = TRUE: A communication partner is expected to establish the connection. If no IP address is entered for a device configured "PASSIVE", this device responds to the first connection request that comes in. If an IP address is entered, the device only responds to the request of the device with the address specified.
ACTIVE	If EN_C = TRUE: The Inline Controller establishes the connection.
PORT=xxxx	Port address of the communication partner (decimal, 1 ≤ port ≤ 65535)
IP=www.xxx.yyy.zzz	IP address of the communication partner. If no IP address is entered in "PASSIVE" mode, the connection to the first partner that logs on is accepted.
CONTO=xxxx	Connect timeout (in ms) The default value for connection establishment confirmation is 3000 ms. After this time another attempt to establish a connection (reconnect; close socket/open socket) is automatically carried out. The time can be changed using this parameter.
RCVTO=xxxx	Receive timeout (in ms) The default value is 5000 ms. If no telegram is received within this time, the IP_URCV function block indicates an error.

3.6.2 IP_USEND

This function block is used to transmit data to a communication partner via an IP connection. The IP_USEND block enables transmission of 1460 bytes, maximum. The IP_CONNECT block must be used to establish a communication connection prior to transmitting data.

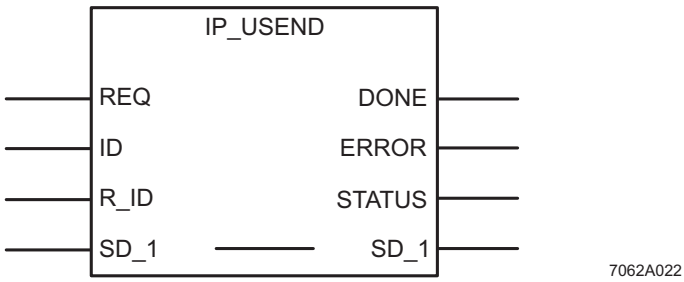


Figure 3-11 IP_USEND block

Table 3-13 IP_USEND function block inputs and outputs

Name	Data Type	Data Direction	Description
REQ	BOOL	Input	Positive edge: Data transmission start. The block is blocked during the entire data transmission process. It cannot be stopped or reset.
ID	INT	Input	During connection establishment using the IP_CONNECT block each communication connection is assigned a unique ID. This ID is used to inform the IP_USEND transmit block to which communication device data is to be transmitted. You can directly connect the ID input to the ID output of the IP_CONNECT function block.
R_ID	STRING	Input	This input is only used for UDP. It is irrelevant for TCP/IP.
DONE	BOOL	Output	This output indicates if the function block successfully and completely sent the data to be transmitted. Successful data transmission can only be evaluated during the cycle with the rising edge at this output.
ERROR	BOOL	Output	Rising edge: An error has occurred. The error code can be read at the STATUS output during the cycle with the rising edge.
STATUS	INT	Output	Error code of the error occurred.
SD_1	ANY	Input/output	Block transmit buffer. The variables at transmit buffer SD_1 must have the same size as the variables at receive buffer RD_1 of the corresponding IP_URCV block. The IP_USEND block enables transmission of 1460 bytes, maximum. The number of bytes to be transmitted is entered in the first word of the SD_1 transmit buffer (observe Intel format). This data word is not transmitted. Data is transmitted in the other words. Data is transmitted as byte stream. There is no Intel/Motorola conversion.

3.6.3 IP_URCV

This function block is used to receive data from a communication partner via an IP connection. The IP_URCV function block enables reception of 1460 bytes, maximum.

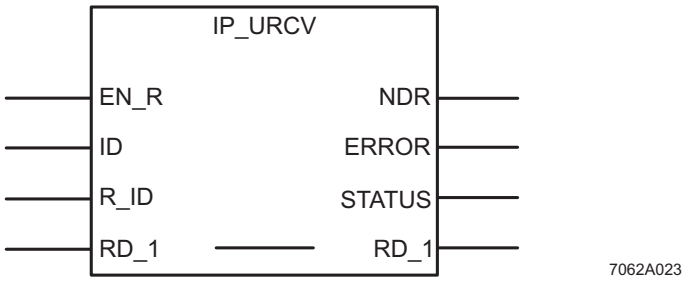


Figure 3-12 IP_URCV block

Table 3-14 IP_URCV function block inputs and outputs

Name	Data Type	Data Direction	Description
EN_R	BOOL	Input	This input is used to prepare the block for data reception. TRUE Data can be received. FALSE Data cannot be received.
ID	INT	Input	During connection establishment using the IP_CONNECT block each communication connection is assigned a unique ID. This ID is used to inform the IP_URCV receive block from which communication device data is to be received. You can directly connect the ID input to the ID output of the IP_CONNECT function block.
R_ID	STRING	Input	This input is only used for UDP. It is irrelevant for TCP/IP.
NDR	BOOL	Input	This output indicates if the function block successfully and completely received the data to be transmitted. Successful data transmission can only be evaluated during the cycle with the rising edge at this output.
ERROR	BOOL	Output	Rising edge: An error has occurred. The error code can be read at the STATUS output during the cycle with the rising edge.
STATUS	INT	Output	Error code of the error occurred.
RD_1	ANY	Input/output	Block receive buffer. The variables at receive buffer RD_1 must have the same size as the variables at transmit buffer SD_1 of the corresponding IP_USEND block. The IP_URCV function block enables reception of 1460 bytes, maximum. The firmware enters the number of data received in the first word of the receive buffer. Data is received in the other words. As long as the input EN_R = TRUE, receiving of data is possible. If a data block has been received, the NDR output is set for the period of one data cycle.

3.6.4 Error Codes

The following error codes are available at the STATUS output. Not all error codes apply to each function block.

Table 3-15 Error codes for TCP/IP at the STATUS output

Error Code		Description
ERR_TRANS_NO_SOCKET	0010_{hex}	Socket initialization failed.
ERR_CONNECTION_LOST	0031_{hex}	The connection was aborted.
ERR_TIMEOUT	0033_{hex}	General timeout, receiver is no longer responding or transmitter has not completed transmission.
ERR_STRING_TOO_LONG	0040_{hex}	Partner string on IP_CONNECT block is too long (255 characters, maximum).
ERR_INVALID_IP	0041_{hex}	The specified IP address is invalid or could not be interpreted correctly.
ERR_INVALID_PORT	0042_{hex}	Port number not permissible.
ERR_NO_FREE_ CONNECTION	0053_{hex}	All available connections are being used.
ERR_FB_EXISTS	0065_{hex}	A different block instance already operates on this connection.

4 System Variables and Status Information

4.1 System Variables

4.1.1 General Information

This section describes the special program functions of the PC WorX software that are available for the Inline Controller.

The Inline Controller has a register set, which is used for diagnostics and easy control of the bus system. The diagnostic data is stored in the diagnostic status register and the diagnostic parameter register. These registers are available to the application program as system variables (system flags, global variables).

Operating states, error states, and additional information about the INTERBUS system can be evaluated in the application program.

For additional information on diagnostics, please refer to the following user manual:

– INTERBUS Diagnostics Guide
  IBS SYS DIAG DSC UM E Order No. 2747293

4.1.2 Status Register for Local Digital Inputs and Outputs

The following system variables can be used to read the local digital input and output states and to write the local digital output states.

Table 4-1 System variables of the status register for local digital inputs and outputs

System Variable	Type	Meaning
ONBOARD_INPUT	WORD	State of all local inputs
ONBOARD_INPUT_BIT0	BOOL	State of local input IN1
ONBOARD_INPUT_BIT1	BOOL	State of local input IN2
ONBOARD_INPUT_BIT2	BOOL	State of local input IN3
ONBOARD_INPUT_BIT3	BOOL	State of local input IN4
ONBOARD_INPUT_BIT4	BOOL	State of local input IN5
ONBOARD_INPUT_BIT5	BOOL	State of local input IN6
ONBOARD_INPUT_BIT6	BOOL	State of local input IN7
ONBOARD_INPUT_BIT7	BOOL	State of local input IN8
ONBOARD_INPUT_BIT8	BOOL	State of local input IN9
ONBOARD_INPUT_BIT9	BOOL	State of local input IN10
ONBOARD_INPUT_BIT10	BOOL	State of local input IN11
ONBOARD_INPUT_BIT11	BOOL	State of local input IN12
ONBOARD_OUTPUT_BIT0	BOOL	State of local output OUT1
ONBOARD_OUTPUT_BIT1	BOOL	State of local output OUT2
ONBOARD_OUTPUT_BIT2	BOOL	State of local output OUT3
ONBOARD_OUTPUT_BIT3	BOOL	State of local output OUT4
ONBOARD_OUTPUT_OVERLOAD_0_3	BOOL	One local output overloaded

4.1.3 Status Register of an I/O Controller (Only ILC 350 PN)

Table 4-2 Status register variables of an I/O controller

System Variable	Type	Meaning
PNIO_CONFIG_STATUS	WORD	Configuration status of the I/O controller
PNIO_CONFIG_STATUS_READY	BOOL	The variable is set if the I/O controller was initialized without errors. Up to now, no desired configuration has been loaded by PC WorX.
PNIO_CONFIG_STATUS_ACTIVE	BOOL	The variable is set if the desired configuration for the I/O controller has been loaded. In this state the I/O controller attempts to cyclically establish a connection to all devices of the desired configuration (under the PROFINET symbol). As for the inaccessible devices: Cyclically at an interval of approx. 5 ms, the I/O controller repeatedly attempts to establish a connection. A connection cannot be established, e.g., if the corresponding device is ready to operate but no correct device name has yet been assigned to it.
PNIO_CONFIG_STATUS_FAULT	BOOL	The variable is set if an error occurred during download/evaluation of the desired configuration.
PNIO_FORCE_FAILSAFE	BOOL	Setting this variable sets all the application relationships to a safe state.

4.1.4 INTERBUS Register of the ILC 350 PN

There may be several INTERBUS masters in each PROFINET I/O system. This is why the registers of an INTERBUS master are directly available as process data. The naming conventions for the system variables of local INTERBUS masters (as, e.g., with the ILC 350 ETH) are also directly used.

4.1.5 Diagnostic Status Register

Information about the operating state of the Inline Controller is stored in the diagnostic status register. Every bit in the diagnostic status register is assigned a certain Inline Controller state.

The following system variables can be used to read the diagnostic status register information.

Table 4-3 System variables of the diagnostic status register

System Variable	Type	Meaning
MASTER_DIAG_STATUS_REG_USER	BOOL	User error/parameterization error
MASTER_DIAG_STATUS_REG_PF	BOOL	I/O error
MASTER_DIAG_STATUS_REG_BUS	BOOL	Bus error
MASTER_DIAG_STATUS_REG_CTRL	BOOL	Error on the Inline Controller/hardware fault
MASTER_DIAG_STATUS_REG_DTC	BOOL	Diagnostic routine is active
MASTER_DIAG_STATUS_REG_RUN	BOOL	Data transmission is active
MASTER_DIAG_STATUS_REG_ACT	BOOL	Selected configuration ready to operate
MASTER_DIAG_STATUS_REG_RDY	BOOL	Inline Controller ready to operate
MASTER_DIAG_STATUS_REG_BSA	BOOL	Bus segment aborted
MASTER_DIAG_STATUS_REG_SYSFAIL	BOOL	System failure
MASTER_DIAG_STATUS_REG_RES	BOOL	Standard function has been processed negatively
MASTER_DIAG_STATUS_REG_SYNCRES	BOOL	Synchronization error occurred
MASTER_DIAG_STATUS_REG_DCR	BOOL	Faulty data cycles
MASTER_DIAG_STATUS_REG_WARN	BOOL	Defined warning time exceeded
MASTER_DIAG_STATUS_REG_QUAL	BOOL	Defined error density exceeded
MASTER_DIAG_STATUS_REG_SSINFO	BOOL	Pending message
MASTER_DIAG_STATUS_REG_HI	BYTE	Master diagnostic status register, high byte
MASTER_DIAG_STATUS_REG_LOW	BYTE	Master diagnostic status register, low byte

4.1.6 Diagnostic Parameter Register

The diagnostic parameter register provides additional information on the error indicated in the diagnostic status register. The following information is stored in the diagnosticparameter register:

– Error location
– Error code

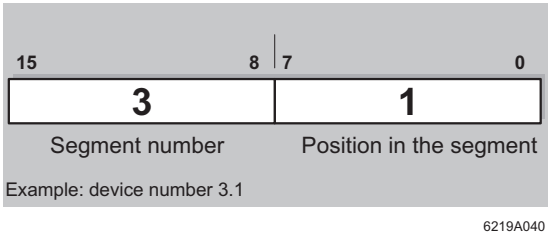


Figure 4-1 Error location in the diagnostic parameter register

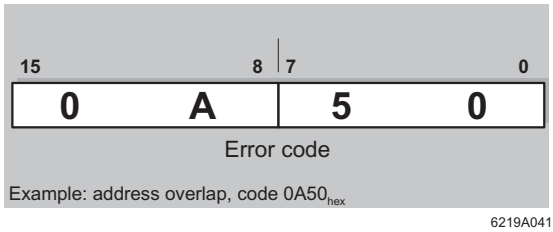


Figure 4-2 Error code in the diagnostic parameter register

Special case: If an interface error cannot be located, the value 128 is indicated in the diagnostic parameter register, i.e., bit 7 is set.

Table 4-4 Error code (example: $0A50_{hex}$)

Error Code:	Type:	Description and Remedy
0A50	USER	Error when assigning a process data item. Input or output address assigned more than once. **Remedy:** Always assign just one process data item to each input/output address. **Add_Error_Info:** PDRL_Index of the incorrect PDRL entry.

The diagnostic parameter register is rewritten whenever an error occurs. The diagnostic parameter register contains value "0" if no errors are detected.

Table 4-5 System variable of the diagnostic parameter register

System Variable	Type	Meaning
MASTER_DIAG_PARAM_REG	WORD	Error location (for a bus error) or error code
MASTER_DIAG_PARAM_REG_HI	BYTE	Diagnostic parameter register, high byte
MASTER_DIAG_PARAM_REG_LOW	BYTE	Diagnostic parameter register, low byte
MASTER_DIAG_PARAM_2_REG_HI	BYTE	Extended diagnostic parameter register, high byte
MASTER_DIAG_PARAM_2_REG_LOW	BYTE	Extended diagnostic parameter register, low byte

4.1.7 IEC 61131 Runtime System

There is a separate group of system variables for the IEC 61131 runtime system.

Table 4-6 System variables of the IEC 61131 runtime system

System Variable	Type	Meaning
PLCMODE_ON	BOOL	PLC status ON: The runtime system on the Inline Controller is ready to operate.
PLCMODE_RUN	BOOL	PLC status RUN: The application program is running.
PLCMODE_STOP	BOOL	PLC status STOP: The application program is currently not running.
PLCMODE_HALT	BOOL	PLC status HALT: The application program was stopped at an unspecified point.
PLCDEBUG_BPSET	BOOL	Breakpoint set: At least one breakpoint has been set in the application program.
PLCDEBUG_FORCE	BOOL	Variable(s) forced: At least one variable is being continuously overwritten (forced) by PROGRAM WORX.
PLCDEBUG_POWERFLOW	BOOL	Powerflow ON: In "Powerflow" mode, you can see which parts of your application program are being processed. This bit indicates whether "Powerflow" mode is active.
PLC_TICKS_PER_SEC	INT	System ticks per second: This variable shows how many pulses the system clock of the Inline Controller delivers per second.
PLC_SYS_TICK_CNT	DINT	Number of system ticks: This variable shows the total number of pulses delivered by the system clock since the last startup.
PLC_TASK_AVAILABLE	INT	Number of available PLC tasks
PLC_TASK_DEFINED	INT	Number of tasks used
PLC_TASK_1	Record, elements = 17	Information on task 1
:	:	:
PLC_TASK_16	Record, elements = 17	Information on task 16

4.1.8 Control Processor

The system variables listed below show the states of the diagnostic status register on the control processor of the Inline Controller.

Table 4-7 System variables of the control processor

System Variable	Type	Meaning
COP_DIAG_STATUS_REG_PON	BOOL	Power ON (COP): The control processor is ready to operate.
COP_DIAG_STATUS_REG_RT_ERR	BOOL	A runtime error (out of real time) has occurred on the control processor.
COP_DIAG_STATUS_REG_FAT_ERR	BOOL	A fatal error has occurred on the control processor. Division by zero, for example, leads to a fatal error.
COP_DIAG_STATUS_REG_WARN	BOOL	A warning has been issued on the control processor.
COP_DIAG_STATUS_REG_FC_RUN	BOOL	Runtime system RUN
COP_DIAG_STATUS_REG_FC_STOP	BOOL	Runtime system STOP
COP_DIAG_STATUS_REG_FC_HALT	BOOL	Runtime system HALT
COP_DIAG_STATUS_REG_FC_LDG	BOOL	Runtime system LOADING
COP_DIAG_STATUS_REG_FC_DBG	BOOL	Runtime system DEBUG
COP_DIAG_STATUS_REG_FC_RDO	BOOL	Runtime system READONLY
COP_DIAG_PARAM_REG	WORD	Diagnostic parameter register of the control processor
COP_DIAG_PARAM_2_REG	WORD	Extended diagnostic parameter register of the control processor

The system variable listed below shows a status information for the control processor of the Inline Controller.

Table 4-8 System variable of the control processor (firmware version 1.1 or later)

System Variable	Type	Meaning
COP_CPU_LOAD_WARNING	BOOL	The load limits of the control processor have been reached.

4.1.9 Battery, Real-Time Clock, Memory Card

Table 4-9 System variables of the battery, real-time clock and memory card

System Variable	Type	Meaning
RTC_BATTERY_LOW	BOOL	Low capacity of the real-time clock battery.
RTC_DATA_INVALID	BOOL	Data of the real-time clock invalid.
FLASHCARD_PRESENT	BOOL	The Phoenix Contact compact flash card is inserted.

4.1.10 Power Supplies

Table 4-10 System variables of the power supplies

System Variable	Type	Meaning
POWER_SUPPLY_MAIN_OK	BOOL	The 24 V main power supply is OK.
POWER_SUPPLY_INPUTS_OK	BOOL	The 24 V power supply for the local inputs is OK.
POWER_SUPPLY_OUTPUTS_0_3_OK	BOOL	The 24 V power supply for the local outputs is OK. (Bits 0 to 3)

4.1.11 Ethernet Status

Table 4-11 System variables of the Ethernet status (firmware version 4.6F/1.11 or later)

System Variable	Type	Meaning
ETH_PORT1_LINK	BOOL	The external Ethernet port is connected.
ETH_PORT1_100MBIT	BOOL	The baud rate of the external Ethernet port is 100 Mbps.
ETH_PORT1_FULL_DUPLEX	BOOL	A full duplex connection is connected to the external Ethernet port.
ETH_PORT2_LINK	BOOL	The internal Ethernet port is connected.
ETH_PORT2_100MBIT	BOOL	The baud rate of the internal Ethernet port is 100 Mbps.
ETH_PORT2_FULL_DUPLEX	BOOL	A full duplex connection is connected to the internal Ethernet port.
ETH_SRV_FTP_ACTIV	BOOL	The Ethernet FTP service is activated.

4.1.12 Mode Selector Switch

Table 4-12 System variables of the mode selector switch

System Variable	Type	Meaning
KEY_SWITCH_RESET	BOOL	The mode selector switch is in the MRESET position.
KEY_SWITCH_STOP	BOOL	The mode selector switch is in the STOP position.
KEY_SWITCH_RUN	BOOL	The mode selector switch is in the RUN position (not supported in the first version of the ILC 350 ETH).
KEY_SWITCH_RUN_PROG	BOOL	The mode selector switch is in the RUN_PROG position.

4.1.13 System Time

Table 4-13 System variables of the system time

System Variable	Type	Meaning
RTC_HOURS	INT	System time (hours)
RTC_MINUTES	INT	System time (minutes)
RTC_SECONDS	INT	System time (seconds)
RTC_DAY	INT	System time (day)
RTC_MONTH	INT	System time (month)
RTC_YEAR	INT	System time (year)

4.2 Status Information of a PROFINET IO Device (Only ILC 350 PN)

The "~PNIO_DATA_STATE" byte that contains the corresponding status bits is available for every PROFINET IO device. Only if this bit is set, an I/O device provides valid data. Please hide non-written bits in the control program by using a link to the corresponding bit mask. These variables can be linked as process data.

Table 4-14 Variables for the status information of an I/O device

Variable	Type	Meaning
~PNIO_DATA_STATE	BYTE	Status of an I/O device
PNIO_IS_PRIMARY	BOOL	Reserved
PNIO_DATA_VALID	BOOL	Only if this bit is set, the I/O device supplies valid data and all other process values are valid.
PNIO_APPL_RUN	BOOL	Application on the I/O device running.
PNIO_NO_DIAG	BOOL	If this bit is set, no device diagnostics is present.

PNIO_DATA_VALID

For PROFINET IO, it might be a standard situation during operation that an I/O device within the network cannot be accessed (e.g., undocking devices).

There may also be applications that cannot be implemented by means of INTERBUS. It can, for example, be ensured that operation of all other I/O devices will not be affected in the event of a device failure.

Therefore the application program must know whether an I/O device supplies valid data or not. For this reason, the "PNIO_DATA_VALID" process data item exists on each device.

PNIO_APPL_RUN

Indication of this bit may be interesting if future control systems are used as I/O devices. In this case, the bit indicates whether the control program runs on the I/O device or not.

PNIO_NO_DIAG

If this bit is set, an additional implementation step of the basic system enables reading the device diagnostics from the I/O device using a special function block, provided that additional information on the error occurred is required in the control program.

5 Technical Data and Ordering Data

5.1 Technical Data

General Data	
Dimensions	182 mm x 141.1 mm x 71.5 mm (75 mm with switch)
Weight	500 g, approximately
Connection data of Inline-connector	
Connection method	Spring-cage terminals
Conductor cross-section	0.2 mm^2 to 1.5 mm^2 (solid or stranded), AWG 26-14

General Supply

Use a **power supply without fall-back characteristic curve** (see Section "Sizing of the Power Supply" on page 2-24).

24 V Main Supply U_M	
Connection method	Spring-cage terminals
Nominal value	24 V DC
Tolerance	-20% / +20%
Ripple	±5%
Current consumption at nominal voltage (typical)	6 mA + 7 mA for each input set
Current consumption at nominal voltage (maximum)	8 A
Continuation	Through potential routing
Safety equipment	
Surge voltage	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 W are short circuited by the input protective diode.
Polarity reversal	Parallel diodes against polarity reversal; in the event of an error the high current through the diodes causes the preconnected fuse to blow.

Provide an external fuse for the 24 V area. The power supply unit must be able to supply 4 times (400%) the nominal current of the external fuse, to ensure that the fuse blows safely in the event of an error.

24 V Segment Supply U_S	
Connection method	Spring-cage terminals
Nominal value	24 V DC
Tolerance	-20% / +20%
Ripple	±5%
Current consumption at nominal voltage (typical)	10 mA + 10 mA for each output set + load
Current consumption at nominal voltage (maximum)	8 A
Continuation	Through potential routing
Safety equipment	
Surge voltage	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 W are short circuited by the input protective diode.
Polarity reversal	Parallel diodes against polarity reversal; in the event of an error the high current through the diodes causes the preconnected fuse to blow.

This 24 V area must be externally protected. The power supply unit must be able to supply 4 times (400%) the nominal current of the external fuse, to ensure that the fuse blows safely in the event of an error.

24 V Supply U_{ILC}	
Connection method	Spring-cage terminals
Nominal value	24 V DC
Tolerance	-15% / +20%
Ripple	±5%
Permissible range	20.4 V DC to 30 V DC
Current consumption at nominal voltage (typical)	200 mA
Safety equipment	
Surge voltage	Input protective diodes (can be destroyed by permanent overload) Pulse loads up to 1500 W are short circuited by the input protective diode.
Polarity reversal	Serial diode in the lead path of the power supply unit; in the event of an error only a low current flows. In the event of an error, no fuse blows in the external power supply unit.

Observe the current consumption of the Inline terminals

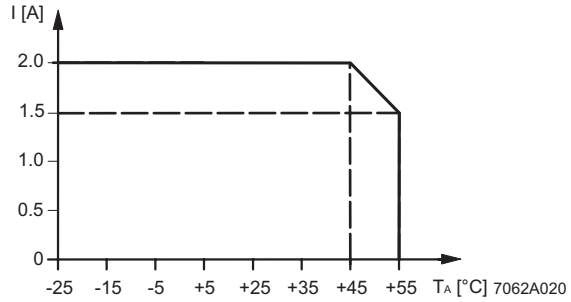
Observe the logic current consumption of each device when configuring an Inline station. It is specified in every terminal-specific data sheet. The current consumption can differ depending on the individual terminal. The permissible number of devices that can be connected therefore depends on the specific station structure.

Protection of the external power supply unit

Ensure protection of 2 A by fuses through the external power supply unit.

7.5 V Communications Power (Potential Jumper)

Nominal value	7.5 V DC
Tolerance	±5%
Ripple	±1.5%
Maximum output current	2 A DC (protected externally)
Derating	



I [A] Logic current in A
TA [°C] Ambient temperature in °C

24 V Analog Supply (Potential Jumper)

Nominal value	24 V DC
Tolerance	-15% / +20%
Ripple	±5%
Maximum output current	0.5 A DC
Safety equipment	Electronic short-circuit protection

INTERBUS

Number of I/O points	8192, maximum
Number of data words	512, maximum
Number of bus segments	254, maximum
Transmission speed	500 kbps or 2 Mbps

This speed is automatically set according to the connected Inline terminals. Only use terminals with a uniform transmission speed in the entire connected Inline system (local bus and remote bus).

Transmission reliability	CR check (hamming distance: 4)
Protocol	EN 50254

Number of Devices in the INTERBUS System

Total number of bus devices	512, maximum
Number of remote bus devices	254, maximum
Number of PCP devices	62
Number of remote bus levels	16, maximum

Network Interface	
Type	Ethernet 10Base-T and 100Base-TX
Speed	10 Mbps (10Base-T), 100 Mbps (100Base-TX) half duplex, full duplex, auto negotiation

This speed cannot be set manually. It is set automatically by means of auto negotiation.

Connection method	CAT5 twisted pair cable RJ45 female connector

Inline Local Bus	
Interface	Inline local bus
Electrical isolation	No
Number of devices	63, maximum, limited by the maximum current consumption from the communications power and analog supply

INTERBUS Remote Bus	
Interface	Inline data jumpers
Electrical isolation	Yes
Number of devices	See technical data for "INTERBUS" and "Number of Devices in the INTERBUS System"

Diagnostic Interface	
Connection method	6-pos. MINI-DIN female connector (PS/2)
Interface type	RS-232
Transmission speed	9600 baud
Electrical isolation	No

PROFINET IO Scope of Functions (ILC 350 PN)	
Device function	PROFINET IO Controller
PROFINET IO specification	Version 1.3
Number of PROFINET IO devices	50, maximum, depending on the update times of the inputs and outputs
Update time of inputs and outputs	1 ms, minimum
Performance specifications	20 PROFINET IO devices, when there is an update time of 8 ms

Digital Inputs	
Number	12
Input design	According to EN 61131-2 Type 1
Definition of switching thresholds	
Maximum low-level voltage	5 V DC
Minimum high-level voltage	15 V DC
Nominal input voltage	24 V DC
Permissible range	-0.5 V < U_{IN} < +30 V DC
Nominal input current at 24 V	7 mA, typical; 15 mA, maximum
Hardware filter times (typical)	
Inputs I1 to I8	
Signal change 0 -> 1 Signal change 1 -> 0	5 µs 2 µs
Inputs I9 to I12	
Signal change 0 -> 1 Signal change 1 -> 0	500 µs 700 µs
Permissible cable length to the sensor	30 m (to ensure conformance with EMC Directive 89/336/EEC)
Use of AC sensors	AC sensors in the voltage range < U_{IN} are limited in application (depending to the input design)

Digital Outputs	
Number	4
Output design	Protected outputs according to EN 61131-2
Nominal output voltage	24 V DC
Nominal output current	500 mA

Diagnostic and Status Indicators	
IEC 61131 runtime system (PLC)	RUN, FAIL
Ethernet (ETH)	LINK, ACT, 100
INTERBUS diagnostics (IL)	RDY/RUN, BSA, FAIL, PF
Digital inputs and outputs	I1 to I12, E, Q1 to Q4
Supply voltages	US, UM, UL

IEC 61131 Runtime System	ILC 330 ETH	ILC 350 ...
Programming system	PC WorX	PC WorX
Speed	0.5 ms for 1 K instructions, typical	0.5 ms for 1 K instructions, typical
Shortest cycle time (for cyclic task)	1 ms	1 ms
Program memory	750 kbyte, 63 K instructions in IL, typical	1 Mbyte, 85 K instructions in IL, typical
Data memory	1,5 Mbytes	2 Mbytes
Memory for retentive data	64 kbytes NVRAM	64 kbytes NVRAM
Number of control tasks	16	16

Real-Time Clock

Precision	1 minute/week, maximum
Power reserve	120 days

Ambient Conditions

Degree of protection	IP20 (EN 60529:1991)
Permissible temperature (operation)	-25°C to +55°C
Permissible temperature (storage/transport)	-25°C to +85°C

This temperature range is only guaranteed if the Inline Controller is mounted horizontally.

Permissible humidity (operation/storage/transport)	10 % to 95 % according to DIN EN 61131-2
Permissible air pressure (operation/storage/transport)	70 kPa to 106 kPa (up to 3000 m above sea level)
Vibration	5g

Conformance With EMC Directive 89/336/EEC

Noise Immunity Test According to EN 61000-6-2

Electrostatic discharge (ESD)	EN 61000-4-2/ IEC 61000-4-2	Criterion B 6 kV contact discharge 8 kV air discharge
Electromagnetic fields	EN 61000-4-3 IEC 61000-4-3	Criterion A Field strength: 10 V/m
Fast transients (burst)	EN 61000-4-4/ IEC 61000-4-4	Criterion B Supply lines: 2 kV Signal/data lines: 2 kV
Surge test	EN 61000-4-5 IEC 61000-4-5	Criterion B Signal/data lines: 1 kV Supply lines: 0.5 kV
Conducted interference	EN 61000-4-6 IEC 61000-4-6	Criterion A Test voltage 10 V

Noise Emission Test According to EN 61000-6-4

Noise emission of housing	EN 55011	Class A

Approvals

The latest approvals can be downloaded at www.download.phoenixcontact.com.

5.2 Ordering Data

5.2.1 Modules

Description	Type	Order No.	Pcs./Pkt
Inline Controller, complete with accessories (connectors and labeling fields)	ILC 330 ETH	2737193	1
Inline Controller, complete with accessories (connectors and labeling fields)	ILC 350 ETH	2737203	1
Inline Controller, meets the demands of the Germanischer Lloyd (German Lloyd) and the Lloyds Register, complete with accessories (connectors and labeling fields)	ILC 350 ETH/M	2985819	1
Inline Controller for PROFINET, complete with accessories (connectors and labeling fields)	ILC 350 PN	2876928	1

5.2.2 Accessories

Description	Type	Order No.	Pcs./Pkt
Parameterization memory (compact flash card) 32 MB	IBS CF FLASH 32MB	2737070	1
Parameterization memory (compact flash card) 64 MB	IBS CF FLASH 64MB	2737054	1
Connecting cable for connecting the Inline Controller to a PC (RS-232 cable)	PRG CAB MINI DIN	2730611	1
QUINT POWER power supplies	See latest INTERFACE catalog from Phoenix Contact		

5.2.3 Software

Documentation

Description	Type	Order No.	Pcs./Pkt
PC WorX automation software	See latest AUTOMATIONWORX catalog from Phoenix Contact		
Configuration and diagnostic software for Ethernet networks in the automation environment (Factory Manager)	FL SWT	2831044	1

5.2.4 Documentation

Description	Type	Order No.	Pcs./Pkt
"Configuring and Installing the INTERBUS Inline Product Range" user manual	IB IL SYS PRO UM E	2743048	1
"Automation Terminals of the Inline Product Range" user manual	IL SYS INST UM E	2698737	1
INTERBUS Diagnostics Guide	IBS SYS DIAG DSC UM E	2747293	1
PC WorX 3 Quick Start Guide	PC WORX 3 QS UM E	2828727	1
Quick Start Guide for PC WorX version 5.00 or later	UM QS EN PC WORX	2699862	1

A Technical Appendix: Service and Maintenance

A 1 Error Causes and Remedies

Table A-1 Installation error causes and remedies

Error	Cause	Remedy
No outputs can be set.	– The voltage supply U_S is absent (see diagnostic LED)	– Connect the voltage supply.
The devices connected to the Inline Controller cannot be read.	– The voltage supply U_S is absent (see diagnostic LED)	– Connect the voltage supply.
The IEC 61131 program is not running.	– Mode selector switch in STP position.	– Set mode selector switch to RUN/ PROG position.
The serial interface is not operating.	– The connector pin assignment of the connecting cable or of the connector adapter used is incorrect.	– Use the PRG CAB MINI DIN connecting cable for the Inline Controller, Order No. 2730611.
The devices on the remote bus cannot be started up.	– The IBS IL 24 RB-T (-PAC) Inline terminal is not installed directly after the Inline Controller.	– Insert the terminal as the first Inline terminal directly next to the Inline Controller.
The diagnostic LED of a device is flashing quickly.	– The device has not been assembled correctly.	– Check the connection to the previous module (bus contacts).

A 2 Updating the Inline Controller Firmware

The firmware (integrated software on the Inline Controller) can be updated using the Ethernet interface. Such firmware updates are exclusively used for adding new functions that are implemented within the scope of continuous product improvement. No firmware update is required for normal system operation.

Please contact your nearest Phoenix Contact representation, if necessary.

A 3 Connecting Unshielded Cables

Unshielded cables are used to connect the I/O devices and the supply voltage to the Inline Controller.

Connect these cables to the relevant Inline connectors using the spring-cage connection method. You may connect cables with diameters of 0.2 mm^2 to 1.5 mm^2.

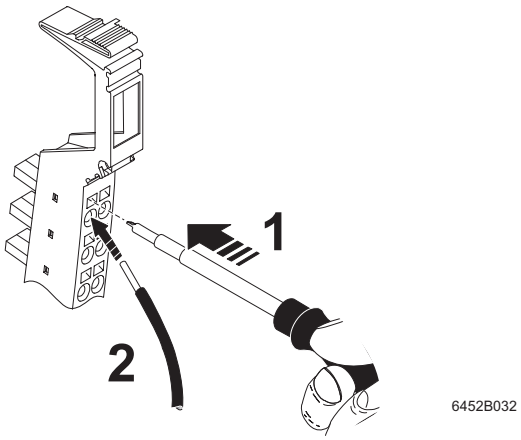


Figure A-1 Connecting unshielded cables

Wire the connectors as required for your application.

When wiring, proceed as follows:

- Strip 8 mm off the cable.

Inline wiring is normally done without ferrules. However, it is possible to use ferrules. If using ferrules, make sure they are properly crimped.

- Push a screwdriver into the slot for the appropriate connection (Figure A-1, detail 1), so that you can plug the wire into the spring opening.
  Phoenix Contact recommends the SFZ 1-0,6x3,5 screwdriver (Order No. 1204517).
- Insert the wire (Figure A-1, detail 2). Remove the screwdriver from the opening. This clamps the wire.

After installation, the wires and the terminal points should be labeled.

B Index

B

Basic wiring of an output 2-30
BootP 3-1
BootP server 2-19

C

Card holder 2-5
Communication path 2-20
Connecting cables A-2
Connecting unshielded cables A-2
Connection elements 2-5
Connectors 2-5, 2-17
Control box 2-16
  See also terminal box
Control cabinet 2-16

D

Default upon delivery 3-1
Diagnostic indicators 2-6
Digital inputs 2-28
Digital outputs 2-28
DIN rail 2-16

E

Electronics base 2-5, 2-17
End clamp 2-16
End plate 2-5, 2-16
Error causes A-1
Error diagnostics 2-6
Ethernet 2-21
Ethernet connection 2-5
Ethernet interface 2-21

F

Fall-back characteristic curve 2-24
Fields of application 2-2, 2-3
FTP function 2-12

H

Hardware requirements 1-1

I

Inline local bus 2-2, 2-3
INTERBUS 2-23
IP address
  Set 3-1
IP20 2-16

L

Local bus 2-23

M

MAC address 2-19
Mode selector switch 2-5, 2-8
Mounting 2-16, 2-17
Mounting location 2-16
Mounting position 2-16

O

Operating elements 2-5

P

Parameterization memory 2-5, 2-10, 2-19
Power supply 2-24
  Connect 2-25
Power supply without fall-back characteristic curve 2-24, 5-1

R

Real-time clock 3-1
Remote bus 2-23
Removing 2-16, 2-18
Replacing 2-19
Reset button 2-5, 2-9
RS-232 cable 2-22
RS-232 interface 2-5

S

Serial PRG interface 2-22
Sizing of the power supply 2-24
Sizing the power supply 2-24
Software requirements 1-1
Status indicators 2-6

System variables 4-1

T

Terminal box 2-16

U

Updating the firmware..... A-1
Upgrade 1-1, 3-1